TAMPEREEN
AMMATTIKORKEAKOULU

DATAN VISUALISOINTI D3-JAVASCRIPT-KIRJASTOLLA

Tuominen Markus

Opinnäytetyö
Syksy 2016
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

TUOMINEN MARKUS:
Datan visualisointi D3-JavaScript -kirjastolla

Opinnäytetyö 31 sivua, joista liitteitä 1 sivu
Syksy 2016

Datan visualisointi on nykyaikana tärkeää, sillä tiedon määrä maailmassa on aivan valtava ja lisää tietoa kerätään koko ajan. Opinnäytetyössä tutustutaan datan visualisointiin D3-JavaScript -kirjastolla sekä yleisesti datan visualisointiin, erityisesti internetissä esitettäviin visualisointeihin. Lisäksi tutkitaan myös D3-kirjaston kanssa hyödyllisiä web-teknologioita ja hyvin yhteensopivia JavaScript-kirjastoja.

D3-kirjaston ominaisuuksia ja toimintaa selvitettiin tekemällä interaktiivisia visualisointeja kyselytutkimusdatasta. Tarkoitus oli selvittää, että mitä D3-kirjastolla pystyy tekemään ja millainen se on, kun sitä vertaa muihin visualisointityökaluihin.

Visualisointi D3-kirjastolla oli aluksi melko vaikeaa, mutta oppimista helpotti aikaisempi osaaminen web-teknologioista. D3-kirjasto vastaa hyvin nykyajan asettamia vaatimuksia internetissä esitettäville visualisoinneille ja sillä pystyi toteuttamaan kaikki mitä yritettiin eli esimerkiksi kyselytutkimusdatassa taustatekijöillä jakaminen, pitkittäistietojen näyttäminen yhdessä kuvaajassa kaiken muun lisäksi ja haluttujen vastausvaihtoehtojen piilottaminen selitteitä klikkaamalla.

D3-kirjasto on erittäin hyvä työkalu moniin visualisointeihin, mutta erityisesti havaittiin sen soveltuvan monimutkaisten datojen visualisointeihin. Monitasoisen datan visualisointi oli D3-kirjastollakin melko vaikeaa, kunnes tajusi kaikkien tärkeimpien funktioiden toiminnot kunnolla, mutta muilla työkaluilla se olisi ollut vielä vaikeampaa ja useimmilla jopa mahdotonta. D3-kirjasto tarjoaa työkalut juuri sellaisten kuvaajien ja muiden visualisointien tekemiseksi kuin itse haluaa, jolloin kaikki on vain mielikuvituksesta ja osaamisesta kiinni.

Asiasanat: datan visualisointi d3 d3.js d3-kirjasto javascript

ABSTRACT

Tampere University of Applied Sciences
ICT Engineering
Software Engineering

TUOMINEN MARKUS:
Data Visualization with D3 JavaScript Library

Bachelor's thesis 31 pages, appendices 1 page
Autumn 2016

Data visualization is important nowadays because the amount of information in the world is enormous and it just keeps growing. The purpose of this thesis is to explore data visualization with D3 JavaScript library and data visualization in general, especially when it comes to visualizations that are presented in the internet. Other useful or mandatory web technologies to use in conjunction with D3 library are also explored as are well compatible JavaScript libraries.

D3 library's properties and functions were researched by doing interactive visualizations with survey data. The intention was to figure out what is able to be done with D3 library and how it compares to other visualization tools.

Visualization with D3 library was difficult at first but the learning process was made easier by previous knowledge about web technologies. D3 library satisfies the needs of today's data visualizations that are created for the internet and it was able to be used for everything it was tried to be used for, e.g. dividing survey data by a background variable, showing longitudinal data in one graph along with everything else and hiding answer options by clicking their legends.

D3 library is a great tool for a lot of data visualizations but it was especially well suited for visualization of complex data. Visualization of multi-level data was difficult even with D3 library until all of the most important functions were really understood well but with other tools it would have been even harder and even impossible with a lot of them. D3 library offers tools to create any kind of graphs or other visualizations as one would want to create and then it's just about imagination and know-how.

Key words: data visualization d3 d3.js library javascript

SISÄLLYS

1 JOHDANTO 7
2 DATAN VISUALISOINTI 8
2.1 Yleistä 8
2.2 Historia 8
2.3 Hyvät periaatteet 9
3 D3-JAVASCRIPT -KIRJASTO 12
3.1 Yleistä 12
3.2 Historia 13
3.3 Erilaisten kuvaajien piirtäminen 13
3.4 D3-kirjasto verrattuna muihin web-pohjaisiin visualisointi-työkaluihin 15
3.5 Uudet ominaisuudet versiossa 4 15
3.6 Vaatimukset datalle 16
4 MUUT D3-KIRJASTON KANSSA HYÖDYLLISET TEKNOLOGIAT 17
4.1 CSS3-tyylimäärittely 17
4.2 HTML5-merkintäkieli 17
4.3 jQuery-JavaScript -kirjasto 18
4.4 Pienemmät JavaScript-kirjastot 18
4.4.1 d3tip-kirjasto 18
4.4.2 Hammer-kirjasto 19
4.4.3 underscore-kirjasto 19
4.4.4 saveSvgAsPng-kirjasto 19
5 ESIMERKKITAPAUS VISUALISOINNISTA D3-KIRJASTOLLA 20
5.1 Yleistä 20
5.2 Data 20
5.3 Interaktiivisuus ja taustatekijöillä jakaminen 20
5.4 Pitkittäistietojen näyttäminen 22
5.5 Kuvaajan ohjelmointi lyhyesti 23
5.5.1 Datan hakeminen ja käsittely 23
5.5.2 Kuvaajan luomisen ensimmäiset vaiheet 24
5.5.3 Kuvioiden ja muiden elementtien lisääminen kuvaajaan 24
6 DATAN VISUALISOINNIN JA D3-KIRJASTON TULEVAISUUS 27
6.1 Yleisesti 27
6.2 D3-kirjaston tulevaisuus 27
6.3 Explorable explanations -käsite 28
7 POHDINTA 29
LÄHTEET 30

LIITTEET 31
Liite 1. Projektin rakenne 31

LYHENTEET JA TERMIT

JavaScript	Web-ohjelmointikieli, jolla voidaan lisätä dynaamista toiminnallisuutta nettisivuille
CSV	Tiedostomuoto, jossa datan arvojen erottamiseen on käytetty pilkkua, tulee sanoista Comma-separated values
TSV	Tiedostomuoto, jossa datan arvojen erottamiseen on käytetty sarkainta, tulee sanoista Tab-separated Values
AJAX	Asynchronous JavaScript and XML, jota käytetään kutsujen tekemiseen palvelimelle tai tietokantaan
HTML	Hypertext Markup Language eli hypertekstin merkintäkieli on internetsivujen kirjoittamiseen käytetty kieli
CSS	Cascading Style Sheets tarkoittaa tyyliohjeita, joilla määritellään internetsivujen tyylit
GitHub	Verkkosivusto versionhallinnalle
Responsiivinen	Sivun elementit mukautuvat käyttäjät selaimen kokoon
Media query	CSS-sääntö, jonka avulla voidaan muokata tyylejä käyttäjän selaimen ja laitteen mukaan
DOM	Document Object Model eli dokumenttioliomalli on ohjelmointirajapinta esimerkiksi HTML-dokumentissa, jossa objektit ovat puudiagrammin mukaisesti

1 JOHDANTO

Opinnäytetyön päätarkoitus on tutkia datan visualisointia ja sen mahdollisuuksia verkossa nykyaikaisilla työkaluilla, tarkemmin sanottuna D3-JavaScript -kirjastolla. D3 on JavaScript-ohjelmointikielen avoimen lähdekoodin kirjasto, jonka avulla voidaan helposti ja nopeasti piirtää erilaisia visualisointeja datan pohjalta. D3-kirjaston pidempi nimi on Data-Driven Documents eli suomennettuna dataohjautuvat dokumentit, joka käytännössä tarkoittaa, että kirjaston tekemät muutokset dokumenttiin pohjautuvat dataan.

Työn aikana tehtävän projektin on tarkoitus opettaa tekijälle D3-kirjaston käyttämistä. Projektin tarvitsemia taitoja opetellaan myös työpaikalla, mutta se ei toimi tilaajana ja työ tehdään yksin. Tärkeää projektissa on saada aikaiseksi toimivia kuvaajia erityisesti moniulotteisesta datasta, jollaista on esimerkiksi kyselytutkimusdata.

Työssä perehdytään myös yleisesti datan visualisoinnin teoriaan, historiaan sekä tulevaisuuteen. Nykyaikana tietoa kerätään enemmän kuin koskaan ennen, joten hyvien visualisointien tekeminen on tärkeää tai muuten tiedon keräämisen tärkeydestä menetetään iso osa, jos tietoa ei ymmärretä kunnolla.

Kaikki tässä työssä esitetyt itse tehdyt kuvaajat on tehty D3-kirjaston versiolla 3, koska versio 4 on hyvin uusi, joten siinä on vielä liikaa bugeja ja lisäksi siinä toteutetaan osa asioista eri tavalla, kuin mitä olen jo ehtinyt toteuttaa ennen uuden version julkaisua. Pääosin versiot ovat kuitenkin ominaisuuksiltaan ja toiminnoltaan hyvin samanlaiset. Uuden version tuomat edut ja haitat käydään läpi kappaleessa 3.5.

D3-kirjaston käyttäminen vaatii osaamista myös muista teknologioista, erityisesti web-teknologioista, jotta sitä voi käyttää tehokkaasti. Työssä tutustutaan näistä oleellisimpiin.

2 DATAN VISUALISOINTI

Tässä luvussa perehdytään datan visualisointiin käsitteenä, datan visualisoinnin historiaan ja visualisoinnin hyviin periaatteisiin. Luvun tarkoitus on siis alustaa projektin lähtökohdat teorian kannalta.

2.1 Yleistä

Datan visualisoinnilla tarkoitetaan datan esittämistä graafisessa muodossa eli esimerkiksi kuvaajan avulla. Yleisesti tämä on tarkoittanut staattista kuvaa esimerkiksi sanomalehdessä. Internetin myötä mahdollisuudet ovat kuitenkin kasvaneet lähes rajattomiksi. Datasta voidaan tehdä kuvaajia, joista käyttäjä voi valita mitä siinä näytetään ja miten se näytetään tai voidaan luoda animaatioita, joissa datan esitystapa kuvaajassa muuttuu tuoden selkeästi esille jonkin datasta havaittavan asian.

Tärkein syy visualisoida dataa on ihmisaivojen tapa prosessoida tietoa: on paljon helpompi ymmärtää suuri määrä monimutkaista dataa kuvaajien avulla kuin laskentataulukon avulla. Lisäksi informaatiota tulee nykyään vastaan kaikkialla ja koko ajan, joten keskittymiskyky yhteen asiaan on lyhyempi kuin ennen eli on todella tärkeää saada esitettyä tieto nopeasti. Visualisointien avulla tulee selkeämmin esille myös esimerkiksi toistuvat kuviot tai muu tärkeä tietoa, joka raakadataa tutkiessa on vaikea havaita. (Shamas 2015.)

2.2 Historia

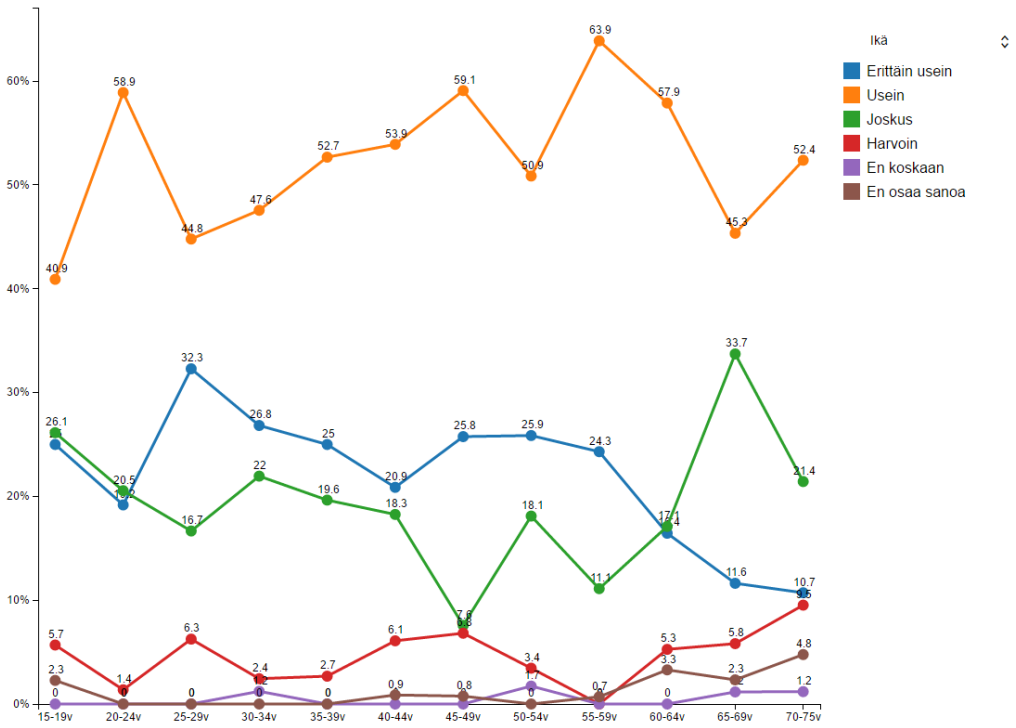
Datan visualisoinnin idea on vanha, sillä karttahan on maantieteellisen datan esittämistä visuaalisesti ja ensimmäiset kartat ovat 8000 vuotta vanhoja. Muun kuin maantieteellisen datan esittäminen visuaalisesti oli kuitenkin harvinaista ja vasta 1700-luvun lopulla alettiin enemmän visualisoida myös muunlaista dataa. Tämä johtui kuitenkin lähinnä siitä, että 1800-luvulle asti datan kerääminen oli harvinaista, joten tietenkin myös datan visualisointi on tällöin harvinaista. (Thompson 2015.)

Yksi ensimmäisistä visualisoinnin uudistajista oli skotlantilainen William Playfair (1759 – 1823), jonka kaaviot näyttävät hyvin moderneilta ja hän keksikin useita kaaviotyyppejä: viiva-, alue-, pylväs- ja piirakkakaavion. 1800-luvun puoleenväliin mennessä datan visualisointeja tekivät jo useat älyköt ympäri Eurooppaa ja oli syntynyt muun muassa dataan pohjautuvat yhteiskuntatieteet ja epidemioita onnistuttiin tukahduttamaan datavisualisointien avulla. Vuosisadan loppuun mennessä koulutetuille ihmisille oli koko ajan tavallisempaa ajatella asioita tilastojen avulla kuin ennen. (Thompson 2015.)

Viimeisen sadan vuoden ja erityisesti viimeisen 40 vuoden aikana datan lukutaito on kehittynyt hurjasti ja nykypäivänä voidaankin luottaa, että lähes kuka vaan pystyy lukemaan ja jopa tekemään datavisualisointeja. Nykyään ohjelmoinnin avulla pystytään visualisoimaan dataa samalla vauhdilla kuin kirjoitetaan uutisia. (Thompson 2015.)

2.3 Hyvät periaatteet

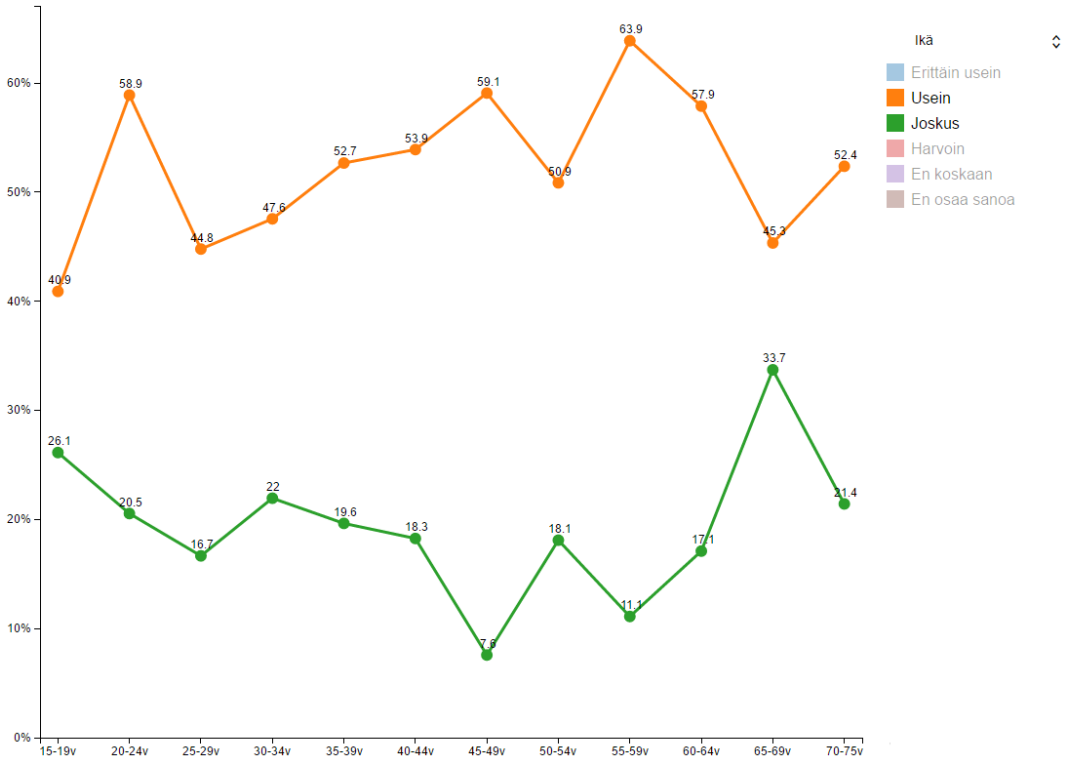
Yksinkertainen on kaunista myös datavisualisoinnissa. Jos yhdessä kuvaajassa on liian paljon tietoa, se on parempi erottaa useaksi eri kuvaajaksi. Liian suuren tiedon määrä yhdellä kuvaajalla näytettäväksi riippuu kuitenkin myös kuvaajan tyypistä ja siitä mitä akseleilla on.



KUVA 1. Mielipidekyselyn vastausten osuudet iän mukaan

Kuvassa 1 näytetään mielipidekyselyn eri vastausvaihtoehtojen osuus iän mukaan jaettuna. Tämä toimii hyvin, sillä viivojen avulla on helppo nähdä miten osuudet muuttuvat iän muuttuessa. Lisäksi on silti helppo nähdä eri vastausvaihtoehtojen osuudet toisiinsa verrattuna: korkeammalla oleva piste tarkoittaa, että useampi on tätä mieltä ja samanaikaisesti oikeammalla oleva piste kertoo vastaajan olevan vanhempi.

Datan visualisointi Internetissä on mahdollistanut kuvaajien interaktiivisuuden, joten onkin hyvä idea antaa käyttäjän jossain määrin valita mitä kuvaajassa näytetään tai millä tarkkuudella data esitetään.



KUVA 2. Kuvan 1 kuvaaja, josta on piilotettu vastausvaihtoehtoja

Kuvassa 2 on sama data pohjalla kuin kuvassa 1, mutta nyt siinä on piilotettu osa vastausvaihtoehdoista, aivan kuten käyttäjät voivat haluta tehdä, jos heitä kiinnostaa vain tiettyjen vastausten osuudet.

Visualisoinnissa tulee aina miettiä mitä sillä halutaan kertoa ja kuinka tarkasti. Lisäksi on hyvä ottaa kohdeyleisö huomioon eli täytyy miettiä mitä kyseiselle kohdeyleisölle halutaan kertoa, jolloin visualisointi saattaa olla erilainen eri kohdeyleisöille, vaikka siinä esitettäisiin sama data. (OCSI 2009.)

Monesti tarkat luvut eivät ole tärkeitä vaan paljon tärkeämpää on tuoda selkeästi esille asia, jota visualisoinnilla oikeasti halutaan näyttää eli esimerkiksi jokin muutos tai selkeä havainto. Lisäksi on tärkeää merkitä hyvin mitä kuvaajassa on eli otsikko ja akseleiden tekstit ovat tärkeitä. Hyvässä datavisualisoinnissa on myös tärkeää saada katsoja kunnolla ajattelemaan visualisoinnissa esitettyä asiaa.

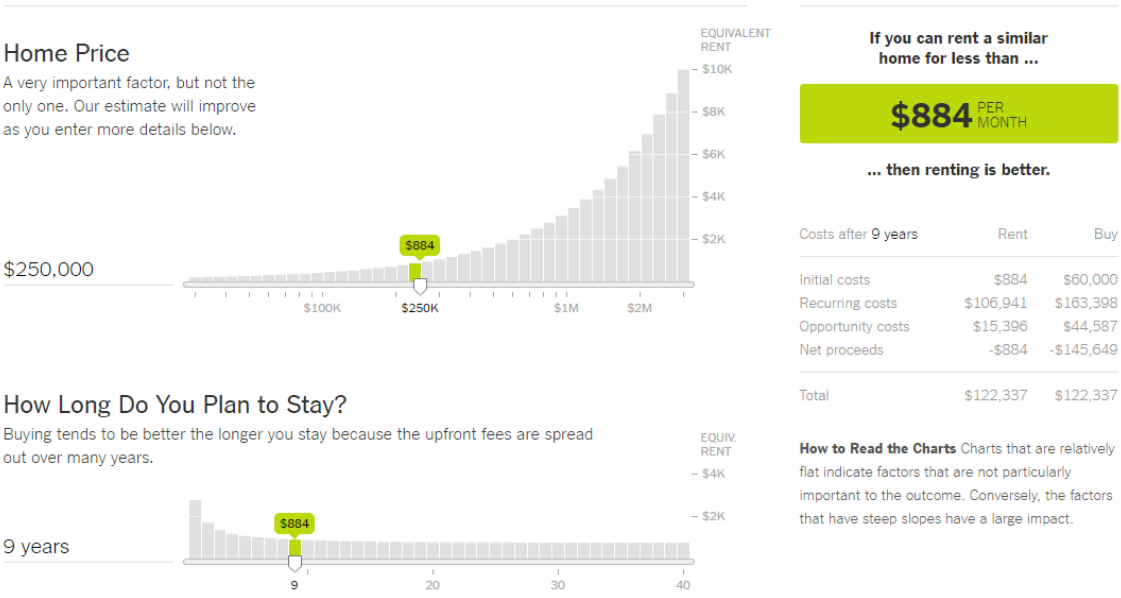
3 D3-JAVASCRIPT -KIRJASTO

Luku käsittelee D3-JavaScript -kirjastoa ensin yleisten asioiden ja historian kannalta. Myöhemmissä kappaleissa D3-kirjaston ominaisuuksiin kuvaajien piirtämisessä ja verrataan sitä kilpaileviin työkaluihin. Lisäksi käydään läpi uusimman version ominaisuuksia ja kirjaston asettamia vaatimuksia käytettävälle datalle.

3.1 Yleistä

D3.js on datan visualisoimista varten tehty JavaScript-kirjasto, mutta monipuolisuutensa vuoksi sitä voi käyttää myös moneen muuhunkin. D3-kirjaston erikoisuutena on, että pystyy tekemään juuri sellaisen kuvaajan kuin haluaa, koska kaikki on aina muokattavissa, yleensä vieläpä helposti ja nopeasti. Sillä pystyy lisäksi luomaan isostakin datasetistä helposti hallittavan layoutin. Lähes minkä tahansa kuvaajiin, kuvioihin, tekstiin, symboleihin, väreihin, selitteisiin tai laskutoimituksiin liittyvän pystyy toteuttamaan D3-kirjastolla. Esimerkiksi myös animaatioiden tekeminen onnistuu.

Yksi hyvä esimerkki D3-kirjaston käyttömahdollisuuksista, kun sen hyvin hallitsee, on The New York Times:n julkaisema interaktiivinen datavisualisointi, jolla voi laskea onko parempi vuokrata vai ostaa, kun eri muuttujien arvoja muuttaa (Bostock, Carter & Tse 2014).



KUVA 3. Kuvankaappaus pienestä osasta “Is it Better to Rent or Buy”-interaktiivista visualisointia (Bostock, Carter & Tse 2014).

Yksi visualisoinnin toteuttajista on Mike Bostock eli D3:n pääkehittäjä, joten ei mikään ihme, että visualisointi on näyttävä, helppokäyttöinen ja helposti ymmärrettävissä.

3.2 Historia

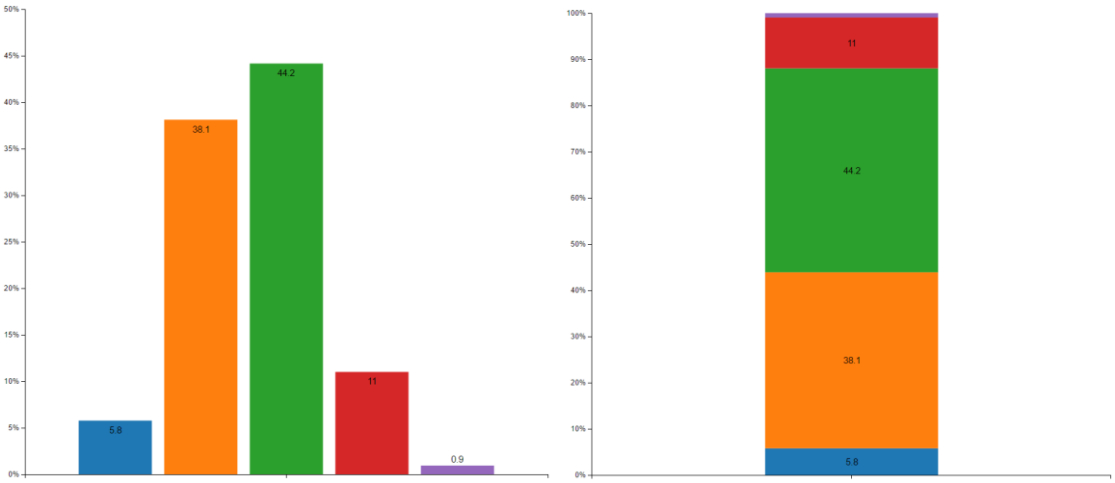
Vuonna 2009 Jeff Heer, Mike Bostock ja Vadim Ogievetsky kehittivät JavaScript-kirjaston nimeltä Protovis, jolla pystyi luomaan SVG-grafiikka datan pohjalta. Vuonna 2011 Protovis-kirjaston kehitys lopetettiin, kun samat kehittäjät alkoivat toteuttaa D3-kirjastoa, joka julkaistiin ensimmäisen kerran kyseisenä vuonna. (Bostock 2012.)

Pääkehittäjänä toimii edelleen Mike Bostock, mutta kirjastoa kehittävät myös useat muut ihmiset ympäri maailman korjaamalla esimerkiksi bugeja tai lisäämällä pieniä toimintoja, mitkä Bostock tarkastaa ja mahdollisesti lisää viralliseen julkaisuun GitHub:n avulla.

3.3 Erilaisten kuvaajien piirtäminen

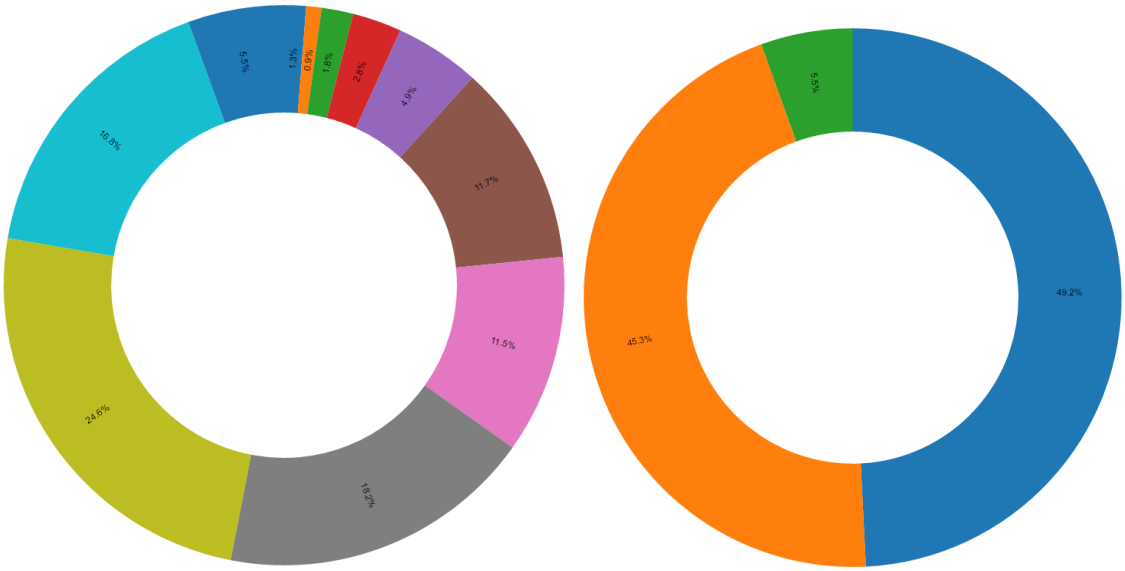
Erilaisille datoille sopivat erilaiset kuvaajat. Lisäksi erilaisilla kuvaajilla saadaan näytettyä datasta ilmeneviä eri asioita helpommin. Esimerkiksi pisteviivakuvaajasta näkee usean vuoden muutokset helposti, kun taas pinotuista vaakapylväistä on helpompi vertailla eri osuuksien määrää toisiinsa nähden.

D3-kirjastolla erilaisten kuvaajien tekeminen on yksinkertaista: kun osaa tehdä yhdenlaisen kuvaajan, niin osaa tehdä myös muunlaiset. Niiden kaikkien pohjana toimivat samanlaiset yleiset kuvaajafunktiot sekä pieni määrä niille spesifejä funktioita ja attribuutteja. Näiden funktioiden ja attribuuttien toiminnot vastaavat kuitenkin toisiaan ja niiden erot ovat helppo ymmärtää. Esimerkiksi ympyrälle syötetään säde, kun taas suorakulmiolle syötetään leveys ja korkeus.



KUVA 4. Ryhmitellyt pylväät ja pinotut pylväät

Pylväskuvaajan voi piirtää monella tavalla: pylväät voivat olla ryhmitellyt tai pinotut, ne voivat olla pystysuunnassa tai vaakatasossa ja samassa kuvaajassa voi piirtää pylväät useammasta asiasta, esimerkiksi ristiintaulukoinnin tuloksista. Yhteistä näille on kuitenkin, että niistä näkee helposti jonkin asian osuuden kokonaismäärästä, sillä korkeuseroja on helppo vertailla myös silmämääräisesti.



KUVA 5. Kymmenen kategorian ja kolmen kategorian donitsikuvaaja

Donitsikuvaaja on myös yksinkertainen ja helposti ymmärrettävissä, mutta se soveltuu kuitenkin parhaiten silloin, kun kategorioita on vähän. Lisäksi se on hyvä pitää muutenkin yksinkertaisena, sillä esimerkiksi usein käytetty 3d-efekti vain vaikeuttaa kuvaajan tulkintaa.

3.4 D3-kirjasto verrattuna muihin web-pohjaisiin visualisointi-työkaluihin

Muita suosittuja visualisointityökaluja ovat muun muassa Google Charts ja CanvasJS. Molemmat näistä ovat suunniteltu siten, että niillä pystyy toteuttamaan kauniita peruskuvaajia hyvin helposti ja nopeasti. Lyhyellä koodinpätkällä saa ammattimaisen kuvaajan aikaiseksi ja esimerkiksi kuvaajan tyypin vaihtaminen onnistuu vaihtamalla koodissa yhtä sanaa.

Näillä pystyy osittain tekemään samat asiat kuin D3-kirjastolla, mutta suurin ja tärkein ero on monimutkaisemman datan visualisointi sekä kuvaajien muokattavuus. D3-kirjastolla pystyy visualisoimaan esimerkiksi kyselytutkimuksen tuloksia ristiintaulukoituna vastaajan sukupuolen ja iän mukaan, kun taas Google Charts:lla ja CanvasJS:llä tämä ei ole mahdollista. Lisäksi ne ovat huomattavasti vähemmän muokattavissa kuin D3-kirjasto, mutta toisaalta ne ovat helpompia ja nopeampia, jos halutaan luoda yksinkertaisesta datasta peruskuvaaja, joten ymmärrän niiden saaman suosion. CanvasJS:ää käyttävät jossain määrin esimerkiksi Microsoft, Intel, Apple, Samsung ja monet muut.

D3-kirjasto häviää siis tietyissä asioissa, kun sitä verrataan muihin vastaaviin työkaluihin. Sen valttikortti onkin muokattavuus ja hallittavuus. Kuvaajasta voidaan muokata mitä tahansa juuri sellaiseksi kuin itse halutaan ja suuren ja monimutkaisen datankin saa visualisoitua melko pienellä vaivalla.

3.5 Uudet ominaisuudet versiossa 4

Suurin muutos versiossa 4 on, että se on modulaarinen eli koostuu useista pienistä kirjastoista, joista voi käyttää haluamiaan osia eli ei tarvitse välttämättä käyttää kaikkea. Lisäksi modulaarisuuden ansiosta siihen on muiden kehittäjien helpompi rakentaa lisäosia (plugin). Pienempiä muutoksia ovat mm. väreille saa määritettyä läpinäkyvyyden (opacity), parempi oletustyyli akseleille, paljon nimiavaruusmuutoksia ja paljon pieniä muutoksia useisiin osiin kirjastoa.

Yksi suuri muutos on lisäksi, että elementin attribuuttien funktioissa ei pääse käsiksi sen ympäröivään elementtiin (parent element) eikä edes sen järjestysnumeroon. Kehittäjän

mielestä se kannustaa luomaan parempaa ja yksinkertaisempaa koodia esimerkiksi antamalla itse etukäteen alielementeille (child element) tarvittavat tiedot (Bostock 2016). Itse olen käyttänyt tätä kuitenkin paljon, sillä silloin koodi pysyy mielestäni lyhyenä ja yksinkertaisena, mutta eiköhän uuteenkin tapaan tottuisi, vaikka se vaatisikin hieman lisää koodia samojen asioiden toteuttamiseen.

Vanhalla versiolla tehdyn koodin päivittäminen uuden version mukaiseksi on myös mahdollista, mutta se saattaa tarkoittaa useita kymmeniä työtunteja koodimäärän ollessa suuri eli tuhansia rivejä. Lisäksi tällöin jää helposti bugeja koodiin.

Web-teknologiat kehittyvät nykyään nopeasti ja välillä tuntuu, että jopa liian nopeasti, mutta onneksi vanhemmilla versiolla tehdyt sivut ja sovellukset toimivat lähes aina yhtä hyvin samoilla selaimilla kuin uudemmilla versioilla tehdyt. Yleensä uudemmat versiot helpottavat ja nopeuttavat tekemistä ja korjaavat joitakin bugeja, mutta niihin vaihtaminen kesken projektin ei kuitenkaan yleensä ole välttämätöntä.

3.6 Vaatimukset datalle

D3-kirjaston data-attribuutti ymmärtää dataa monenlaisessa muodossa. Lyhyen ja yksinkertaisen datan voi syöttää taulukon (array) avulla, joka on etukäteen määritelty suoraan JavaScript:ssä. Data on kuitenkin yleensä tallennettuna esimerkiksi csv-tiedostossa tai tietokannassa.

D3-kirjasto sisältää funktiot csv- ja tsv-tiedostojen lukemiseen, mutta sen lisäksi funktiolla dsv voi itse määritellä tiedostossa käytetyn erotinmerkin, jos se ei ole pilkku eikä sarkain. Lisäksi voidaan määrittää, miten data luetaan tiedosta eli minkälainen JavaScript-taulukko datan perusteella luodaan.

Tietokannasta data voidaan hakea AJAX:n avulla JSON-muodossa, jolloin sitä voidaan suoraan käyttää D3-kirjaston elementin data-attribuutissa. Datan voi myös hakea jossain muussa muodossa ja muuttaa sen sitten ensin JSON-muotoon, objektiksi tai taulukoksi riippuen datan monimutkaisuudesta, jonka jälkeen sen voi syöttää D3-kirjaston elementin data-attribuuttiin.

4 MUUT D3-KIRJASTON KANSSA HYÖDYLLISET TEKNOLOGIAT

Tämä luku kertoo muista teknologioista, joita D3-kirjaston kanssa on hyvä käyttää. Läpikäytävät teknologiat ovat käytännössä kaikki joko web-teknologioita tai JavaScript-kirjastoja. Kaikkia tässä luvussa esitettyjä teknologioita käytetään työn aikana tehtävässä projektissa.

4.1 CSS3-tyylimäärittely

CSS tarkoittaa tyyliohjetta, jolla html-elementtien tyylejä voidaan muokata hyvin paljon. CSS3 tarkoittaa uusimpia määrittelyjä ja se sisälsi vuonna 2012 jo yli neljäkymmentä moduulia, joista suurin osa on kesken, mutta osa on jo valmiita ja suositeltu käytettäväksi (CSS3 .info 2012). Tärkein näistä D3-kirjaston kannalta on media query, jonka avulla voidaan helposti määrittää eri laitteille ja eri resoluutioille omat tyylit.

Omien tyylien luominen ei ole pakollista, sillä oletustyylit näyttävät jo varsin hienoilta. D3-kirjastolla luotuja elementtien tyylejä voi muuttaa CSS:n ja luokkien avulla aivan normaalisti tai niille voi luotaessa antaa tyyli-attribuutilla suoraan esimerkiksi värin tai fonttikoon. Elementeille luokkien lisääminen ja poistaminen onnistuvat helposti valmiiden attribuuttien avulla. Myös esimerkiksi akseleiden tyyliä voi muokata.

4.2 HTML5-merkintäkieli

HTML on merkintäkieli, jolla kirjoitetaan erityisesti internet-sivut. HTML5 sisältää suuren määrän elementtejä ja attribuutteja, jotka vastaavat nykyaikaisten nettisivujen tarpeita. Lisäksi siihen sisältyy ohjelmointirajapintoja, joita käytetään JavaScript:n avulla.

D3-kirjasto vaatii kaikkien sillä luotavien elementtien pohjaksi SVG-elementin, mutta sen voi lisätä esimerkiksi div-elementtiin tai suoraan sivun runkoon (englanniksi body). Lisäksi HTML tai ainakin jokin automaattisesti HTML-koodia generoiva kieli kuten Jinja2 on osattava, jos haluaa luodut kuvaajat Internetiin.

4.3 jQuery-JavaScript -kirjasto

jQuery on JavaScript-kirjasto, joka helpottaa ja nopeuttaa lähes kaikkea JavaScript-koodausta kuten esimerkiksi elementtien valitsemista, luomista ja poistamista, tapahtumien käsittelyä ja AJAX-kutsujen tekemistä eli sitä voi kutsua yleiskirjastoksi. Se on ylivoimaisesti suosituin JavaScript-kirjasto ja sitä käyttää 71,3% kaikista nettisivuista (W3Techs 2016). Erityisten hyvää jQuery-kirjastossa D3-kirjaston kannalta on, että niiden syntaksit ovat hyvin samankaltaiset.

Nykyään on olemassa myös muita työkaluja, joilla voi toteuttaa samoja asioita ja vielä paljon muutakin, mutta ne ovat enemmän sovelluskehyksiä kuin kirjastoja. D3-kirjaston kannalta tärkeimmät ominaisuudet löytyvät kuitenkin jo jQuery-kirjastosta, joten yleensä ei ole tarvetta käyttää esimerkiksi Angular 2 -sovelluskehystä. jQuery-kirjastolla pystyy hyvin tekemään tarvittavat DOM-muutokset ja AJAX-kutsut.

4.4 Pienemmät JavaScript-kirjastot

JavaScript-kirjastoja on olemassa lukuisia hyödyllisiä, mutta tässä on listattu näistä tämän työn aikana opetellut ja käytetyt kirjastot. Näiden lisäksi on olemassa niin sanottuja polyfill-koodinpätkiä, joilla mahdollistetaan jokin toiminto sellaisissa selaimissa, joissa sitä ei normaalisti ole. Ne eivät siis ole varsinaisesti kirjastoja, mutta niitäkin kannattaa hyödyntää esimerkiksi classList-JavaScript -attribuutti ei ole oletuksena tuettu kaikissa selaimissa, vaikka se on hyödyllinen, koska sillä pystyy helposti lisäämään ja poistamaan elementtien luokkia.

4.4.1 d3tip-kirjasto

Kuvaajiin on hyvä laittaa pieni laatikko kertomaan tietoa, kun vie kursorin elementin päälle tai klikkaa elementtiä kuvaajassa. Helpoin tapa tehdä tämä on d3tip-kirjasto. Se on päivitetty toimimaan D3-kirjaston version 4 kanssa, mutta siitä löytyy edelleen vanha

versio käytettäväksi version 3 kanssa. Sen käyttäminen on tehty hyvin helpoksi ja vapaaehtoisilla parametreilla voi esimerkiksi muuttaa helposti mihin suuntaan kursorista laatikko avautuu.

4.4.2 Hammer-kirjasto

Hammer-kirjaston avulla pystyy hallitsemaan kosketustoimintoja. Kaikkia peruskosketustoimintoja pystyy liittämään haluttuihin elementteihin ja määrittelemään mitä toiminnot tekevät. Lisäksi oletustoimintoja voi halutessaan ottaa pois päältä tai luoda kokonaan omia kosketustoimintoja kuten vaikkapa neljän sormen kosketus. Kirjasto on siis hyödyllinen, kun luodaan interaktiivisia kuvaajia mobiililaitteille. Voidaan esimerkiksi laittaa kuvaajan asetusvaihtoehdot aukeamaan ruudulle, kun sormea liikuttaa oikeasta reunasta vasemmalle ja sulkeutumaan liikuttamalla vasemmalta oikealle missä tahansa näytöllä.

4.4.3 underscore-kirjasto

Datan hallintaa pystyy helpottamaan huomattavasti jQuery-kirjastolla, mutta underscore-kirjasto on vielä sitäkin parempi tähän tarkoitukseen. Käytännössä kaikki mitä haluaa pystyä tekemään JavaScript-taulukoihin tai objekteihin liittyen on mahdollista underscore-kirjastolla. Siinä on esimerkiksi funktiot, joilla voi luoda taulukon objektin avaimista tai arvoista, etsiä kahden taulukon eroavaisuudet tai etsiä objektista arvon perusteella avain, kun yleensä haetaan avaimen perusteella arvo.

4.4.4 saveSvgAsPng-kirjasto

Lyhyt ja yksinkertainen kirjasto, jolla voi tallentaa D3-kirjastolla piirretyn SVG:n png-kuvatiedostoksi. Käyttö on erittäin helppoa, sillä siinä on vain yksi funktio, jolle annetaan pakollisina parametreina ainoastaan tallennettava SVG-elementti ja tiedostonimi. Muut parametrit ovat vapaaehtoisia.

5 ESIMERKKITAPAUS VISUALISOINNISTA D3-KIRJASTOLLA

Luvun päätarkoitus on esitellä projektin sisältöä ja erityisesti lopputuloksia. Lisäksi luvussa käydään samalla läpi hieman teoriaa ja kerrotaan lyhyesti, miten kuvaaja ohjelmoidaan.

5.1 Yleistä

Esimerkkitapauksessa visualisoidaan kyselytutkimusdataa ja sen tavoitteena on esitellä D3-kirjaston ominaisuuksia työssä tehtyjen asioiden avulla. Vastaukset pystytään muun muassa jakamaan taustatekijöiden kuten esimerkiksi iän, sukupuolen, maakunnan tai puoluekannatuksen mukaan. Kuvaajissa ei ole kysymystekstiä, mutta vastausvaihtoehdoilla on geneeriset selitteet, jotta voidaan esitellä niiden toimintaa.

Kuvista nähdään myös selitteille piirretyt neliöt tekstit eteen, joka kertoo vastauksen värin kuvaajassa sekä valintojen eteen piirretyt ympyrät, jotka kertovat mitä on valittu.

Liitteessä 1 näkyy projektin rakenne, jossa näkyy myös Python-kielen py-tiedostoja, sillä sovellus on rakennettu pyörimään Pyramid web-sovelluskehyksen päällä.

5.2 Data

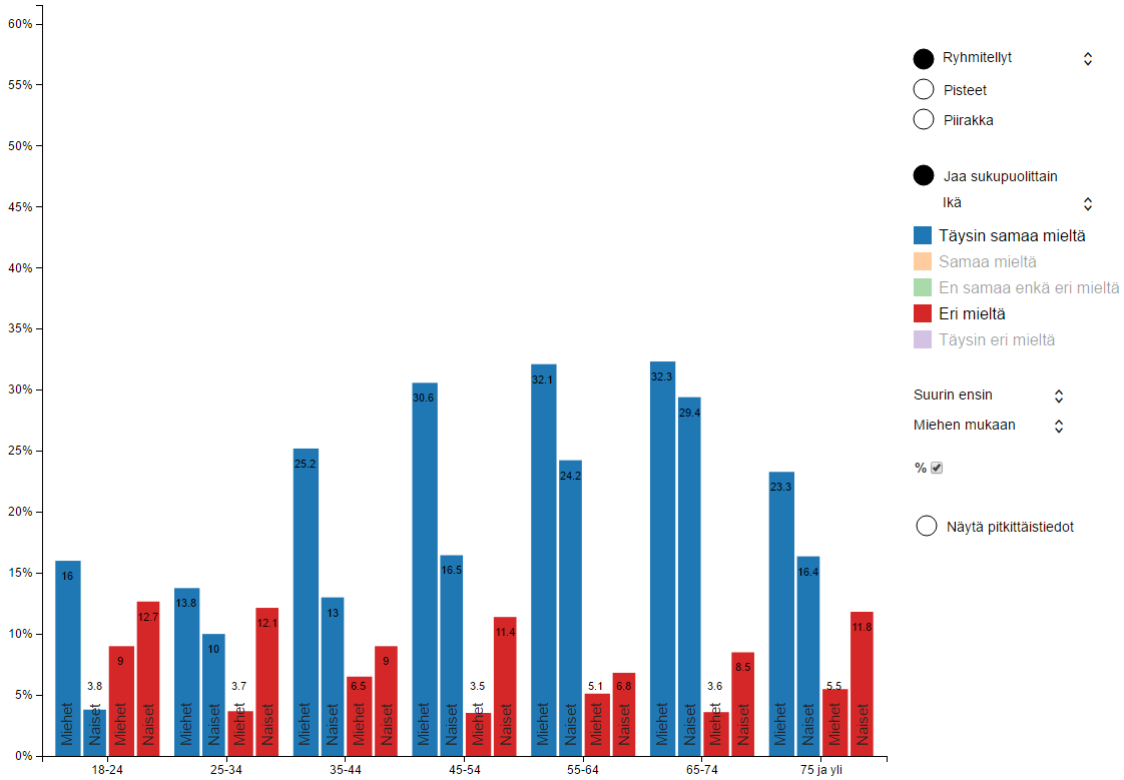
Työssä on generoitu itse koneellisesti kyselytutkimusdataa, joka on tallennettu SQLite-tietokantaan. Tietokannasta data saadaan helposti käytettäväksi JavaScript-koodissa hakemalla se AJAX:n avulla.

5.3 Interaktiivisuus ja taustatekijöillä jakaminen

Interaktiivisuuden lisääminen kuvaajaan on tärkein asia tässä työssä, joten käyttäjälle on annettu mahdollisuus valita lähes kaikki mitä ylipäätään valittavissa voi olla.

Toinen tärkeä asia on taustatekijöillä jakaminen, sillä tällaisen toiminnon tekeminen useimmilla muilla työkaluilla on yksinkertaisesti mahdotonta tai ainakin erittäin vaikeaa, sillä se vaatii monimutkaista datanhallintaa. D3-kirjastossa tähän ratkaisuna on d3.nest-funktio, joka luo haluttujen avaimien perusteella yksi- tai useampitasoisen objektin ja tämä funktio sisältää myös esimerkiksi rollup-attribuutin, jolla voi laskea avaimien määriä ja nekin tallentuvat objektiin.

Lisäksi D3-kirjastossa on erilaisia valmiita layout-funktioita kuten d3.layout.stack-funktio, joihin pystyy syöttämään d3.nest-funktiolla luotuja objekteja, jolloin datan arvojen perusteella luodaan vielä monimutkaisempi objekti, joka sisältää tarvittavat x, y ja y0 arvot kuvaajien piirtämiseen. Kaikissa tämän työn kuvaajissa käytetään jotakin layout-funktiota, yleensä stack-funktiota. Se on käytännössä pakollinen, jos samassa x-akselin pisteessä on päällekkäin useita datapisteitä eli esimerkiksi pinotut pylväät (kuva 4) tai pitkittäistietojen näyttäminen (kuva 7).



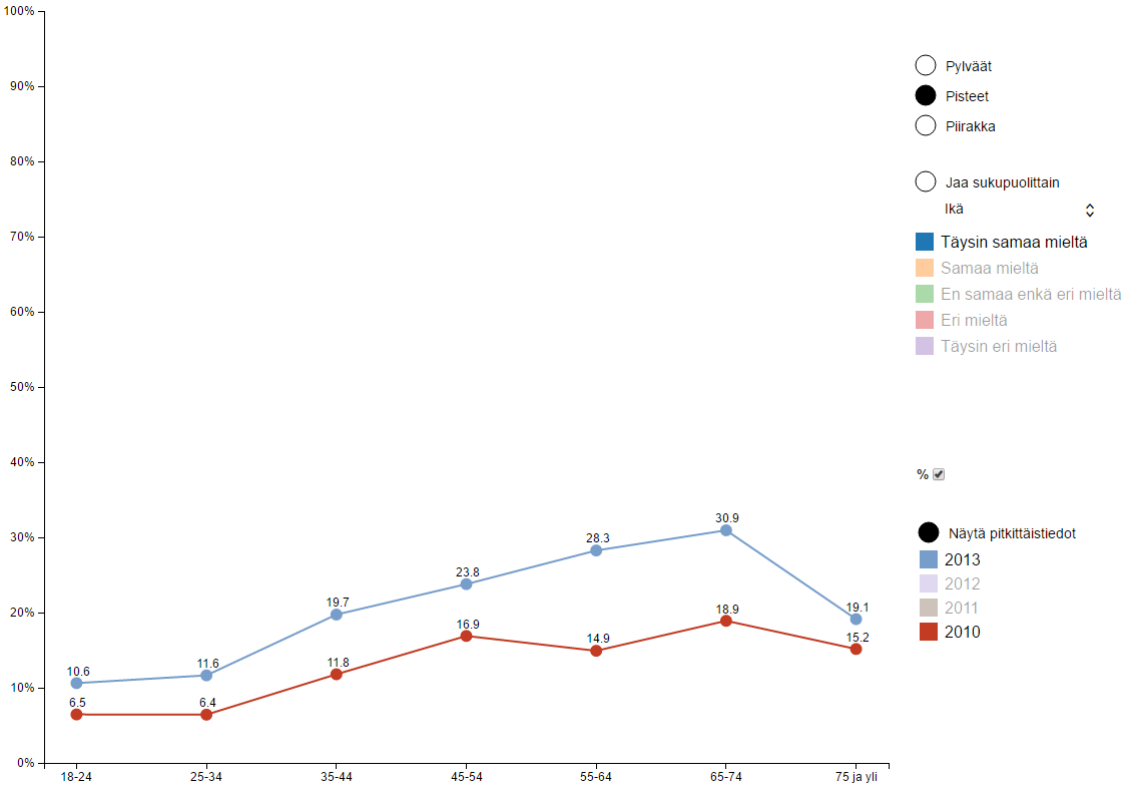
KUVA 6. Kuvaajan asetusvaihtoehdot

Käyttäjälle on annettu vapaus valita kuvaajan tyyppi, jakava taustatekijä, näytettävät vastausvaihtoehdot, järjesteleminen ja prosenttien näyttäminen kuvaajassa. Lisäksi voidaan vielä jakaa kuvaaja aina sukupuolittain, vaikka toinen taustatekijä olisi jo valittuna. Pitkittäistiedot voidaan myös näyttää eli data niiltä vuosilta, kun on kysytty sama kysymys.

Kuvassa 6 näkyy myös miltä kuvaaja näyttä, kun se on samanaikaisesti jaettu kahdella taustatekijällä eli tässä tapauksessa iällä ja sukupuolella. Viidestä vastausvaihtoehdoista on piilotettu kolme. Alasvetovalikko, josta on valittu ”Ryhmitellyt”, antaa mahdollisuuden valita myös pinotut pylväät.

5.4 Pitkittäistietojen näyttäminen

Pitkittäistietoja näytettäessä haasteellista on datan mahdollinen määrä kuvaajassa. Interaktiivisuus antaa käyttäjälle vapaat kädet, joten kuvaajaan on mahdollista saada piirtymään niin paljon viivoja, ettei siitä saa enää mitään selvää. Ei kuitenkaan tuntunut hyvältä ratkaisulta rajoittaa käyttäjän vaihtoehtoja eli vastuu hyvän kuvaajan piirtymisestä on tässä tapauksessa osittain käyttäjällä.



KUVA 7. Pitkittäistiedot kuvaajassa

Pitkittäistietoja näytettäessä annetaan käyttäjälle myös vaihtoehto piilottaa vuosia. Kuvassa 7 on piilotettu vuodet 2012 ja 2011 ja valittu näytettäväksi vain yksi vastausvaihtoehto.

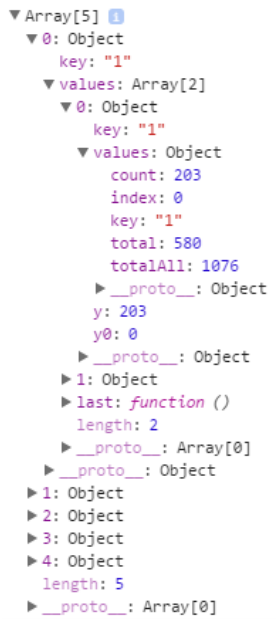
5.5 Kuvaajan ohjelmointi lyhyesti

Tässä luvussa kerrotaan, miten yksittäisen kuvaajan piirtäminen käytännössä tapahtuu, kun perusasiat kuten tarvittavien kirjastojen lisääminen projektiin ja yksinkertaisen HTML-sivupohjan luonti on jo tehty. Piirrettävä kuvaaja on pinotut pylväät, jossa on viisi vastausvaihtoehtoa, vastaukset on jaettu sukupuolen mukaan ja vastausvaihtoehdoista kolme on piilotettu.

5.5.1 Datan hakeminen ja käsittely

Data tarvitaan JSON-muodossa JavaScript-muuttujaan tallennettuna ja se haetaan AJAX:n avulla lähettämällä pyyntö ennalta määrättyyn osoitepolkuun palvelimella. Pyramid-sovelluskehyksen views.py-tiedostoon on määritetty tämä osoitepolku palauttamaan tietokannasta haluttu data valmiiksi JSON-muodossa.

Seuraavaksi data täytyy vielä saattaa sellaiseen muotoon, että sen perusteella on helppo piirtää monipuolinenkin kuvaaja. Data ajetaan ensin nest-funktion läpi, jossa on määritelty, minkä avainten perusteella dataa asetetaan sisäkkäin ja minkä avainten perusteella lasketaan kokonaisarvot. Tämän jälkeen datalle kutsutaan d3.layout.stack-funktio, jotta saadaan tarvittavat arvot avaimille y ja y0. Avaimen y arvo kertoo pylvään korkeuden ja avaimen y0 arvo kertoo miltä korkeudelta piirtäminen pitää aloittaa, jotta pylväät ovat päällekkäin.



KUVA 8. Esimerkki valmiista datasta, jolla voidaan piirtää kuvaaja

Kuvassa 8 näkyy datan rakenne funktioiden ajamisen jälkeen. Jokaisella vastausvaihtoehdolla on lista, joka sisältää objektit kaikille taustatekijän ryhmille, jotka edelleen sisältävät layout-funktion antamia arvoja kuten y ja y0 ja nest-funktion antamia arvoja kuten count ja total.

5.5.2 Kuvaajan luomisen ensimmäiset vaiheet

Kuvaajan piirtämiseen tarvitaan SVG-elementti, joka voidaan luoda D3-kirjaston avulla antamalla sille kohde, johon se luodaan sekä muut tarvittavat attribuutit kuten vähintään leveys ja korkeus. Kohteena käytetään ennalta HTML-tiedostoon kirjoitettua div-elementtiä. Tämän jälkeen SVG-elementtiin voidaan alkaa lisätä elementtejä valmiiden D3-kirjaston funktioiden avulla. Ensimmäisenä täytyy lisätä molemmat akselit eli x-akseli ja y-akseli, jotka tarvitsevat attribuuteikseen muun muassa skaalan ja sijainnin. Skaalan määrittämiseen tarvitaan toinen funktio, jolle pitää määrittää mitä akselilla tulee olemaan ja mitkä ovat akselin rajat.

5.5.3 Kuvioiden ja muiden elementtien lisääminen kuvaajaan

Kuvaajaan lisätään kuvioita luomalla D3-kirjaston avulla jokin kuvio, lisäämällä se luotuun SVG:hen ja syöttämällä sille haluttu data. Datan perusteella kuviota luodaan automaattisesti niin monta kuin datapisteitä on ja attribuuteilla määritellään miten kuviot sijoittuvat kuvaajaan ja miltä ne näyttävät.

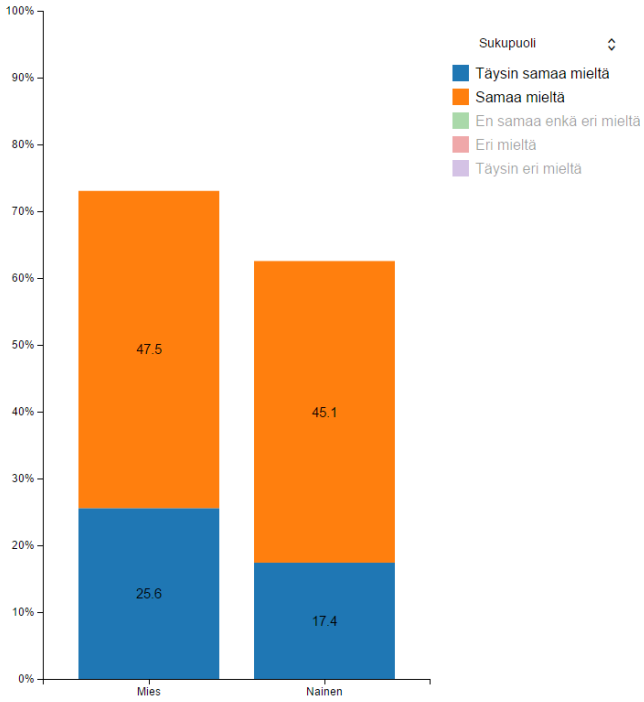
```
var group = svg.selectAll(".group")
    .data(dataset[0])
    .enter().append("g")
    .attr("class", "group");

group.selectAll("rect")
    .data(function (d) { return d.values; })
    .enter().append("rect")
    .attr("id", function (d){ return "rect"+(_.invert(choiceNumbers)[d.values.key])+"_"+d.key; })
    .attr("class", "rects")
    .style("fill", function (d) { return color(_.invert(choiceNumbers)[d.values.key]-1); })
    .attr("y", function (d,i,j) {
        return y(d.y0/originalDataset[0][j].values[d.values.index].values.total) -
            (y(d.y/originalDataset[0][j].values[d.values.index].values.total)-y(0))*-1;
    })
    .attr("height", function (d,i,j) { return (y(d.y/originalDataset[0][j].values[d.values.index].values.total)-y(0))*-1; })
    .data(xDomain)
    .attr("width",function(){ return x.rangeBand(); })
    .attr("x", function (d) { return x(d); });
```

KUVA 9. Koodi, jolla lisätään kuvaajaan pinotut pylväät

Kuvan 9 koodissa luodaan ensin D3-kirjaston ryhmäelementtejä eli g-elementtejä datan perusteella jokaiselle objektille syötetyssä muuttujassa. Sen jälkeen näihin jokaiseen g-elementtiin luodaan suorakulmioita niin monta kuin syötetyössä objektissa on arvoja. Varsinaiseen dataan päästään käsiksi vaikka jokaisen attribuutin kohdalla luomalla funktio ja antamalla sille parametreja. Ensimmäinen parametri viittaa itse dataan, toinen datan indeksiin ja kolmas g-elementin indeksiin, joka sisältää kyseisen suorakulmion.

Suorakulmion luomisen lopussa dataksi vaihdetaan x-akselin data eli tässä tapauksessa sukupuoli, jolloin oikea sijainti x-akselilla saadaan aikaisemmin määritetyn skaalausfunktion x avulla antamalla sille parametriksi tämä data. Leveyteen on valmiita funktioita kuten kuvassa 9 esiintyvä rangeBand, jotka asettavat leveyden luodun skaalan perusteella automaattisesti. Pylvään piirtämisen aloitussijainti y-akselilla on hieman monimutkainen, koska pylväät on piirretty alhaalta ylöspäin eikä vasemmalta oikealle. Joka tapauksessa siinä kuitenkin käytetään hyväksi aikaisemmin luotua skaalausfunktiota y ja datan arvoja y, y0 ja total.

Kuvan 9 originalDataset-muuttujaan tallennetaan data, kun yhtään vastausvaihtoehtoja ei ole piilotettu, jolloin dataset-muuttujasta voidaan poistaa halutut vastausvaihtoehtojen arvot ja piirtää näillä muuttujilla yhdessä oikeanlaiset pylväät.



KUVA 10. Kuvan 8 datalla ja kuvan 9 koodilla luotu kuvaaja

Kuvan 10 kuvaajan luomiseen tarvitaan tietenkin paljon enemmän koodia kuin kuva 9 näyttää kuten esimerkiksi koodi vastausvaihtoehtojen selitteiden luomiseen. Ne pystytään toteuttamaan aivan vastaavasti kuin suorakulmio eli luodaan elementti ja syötetään sille data, joka selitteiden kohdalla tarkoittaa listaa niistä. Suorakulmioille on määritetty class eli luokka-attribuutin arvoksi rects, joten nyt pylväitä pystyisi tyylittelemään määrittelemällä CSS-tiedostossa rects-luokalle tyylit. Kuvassa näkyvä alasvetovalikko on luotu valmiiksi HTML-pohjaan ja datan hakemiseen jälkeen siihen on lisätty vaihtoehdot ja toiminta JavaScript:n avulla.

6 DATAN VISUALISOINNIN JA D3-KIRJASTON TULEVAISUUS

Datan visualisoinnin ja D3-kirjaston tulevaisuutta pystyy päättelemään melko hyvin viimeisimpien vuosien pohjalta. Tässä luvussa keskitytäänkin siihen, mitä viime vuosien perusteella voi sanoa lähitulevaisuudesta. Lisäksi avataan melko uutta explorable explanations -käsitettä.

6.1 Yleisesti

Tiedon määrä maailmassa kasvaa jatkuvasti nopealla tahdilla ja ilman visualisointia kaiken sen tiedon ymmärtäminen on hyvin hankalaa. Tämän takia uskon, että visualisointeja tullaan näkemään vain entistä enemmän joka paikassa ja tarve hyville työkaluille, joilla tehdä näitä visualisointeja tulee kasvamaan. Käytännössä kaikkia eniten käytettyjä työkaluja kehitetäänkin edelleen hyvin nopeaa tahtia.

Nykyään monista asioista kerätään niin paljon tietoa, että siitä on vaikea tehdä visualisointeja, sillä pelkästään jo sen säilyttäminen on haasteellista. Tällaisesta datamäärästä puhutaan nimellä Big Data eli suomeksi suuri data. Big Datan visualisointi on yksi suurimmista datavisualisoinnin haasteista, mutta ei niinkään sen takia, että visualisointityökalut eivät pystyisi visualisoimaan suurta määrää dataa. Haasteet johtuvat siitä, että data voi tulla monesta lähteestä, data voi olla eri muodoissa ja datan määrä kasvaa yleensä todella nopeasti. Dataa saattaa siis joutua käsittelemään paljon jo ennen kuin sitä voidaan visualisoida ja sitten joutuu vielä sen jälkeen miettimään, miten kyseinen data on paras visualisoida.

6.2 D3-kirjaston tulevaisuus

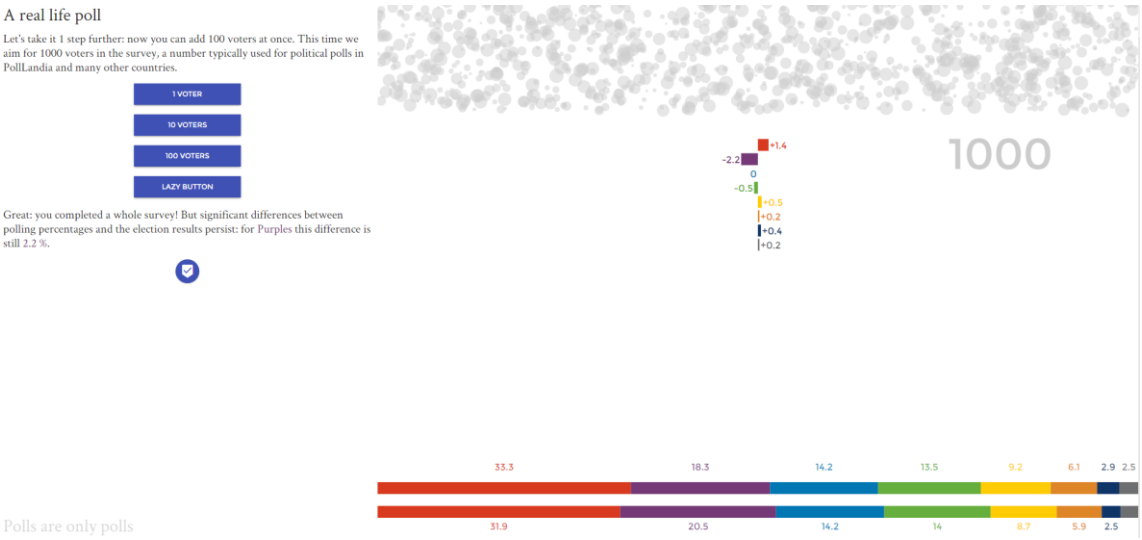
Tällä hetkellä D3-kirjasto ei välttämättä ole paras työkalu esimerkiksi toimittajille, joiden täytyy saada hyvännäköisiä kuvaajia luotua erittäin lyhyessä ajassa. Sen avulla on kuitenkin mahdollista tehdä paljon muuta nykyaikana hyödyllistä visualisointia kuten laajoihin datasetteihin tai monimutkaiseen dataan liittyviä visualisointeja, monipuolisia interaktiivisia ja responsiivisia kuvaajia sekä explorable explanations -tyyppisiä opettavia visualisointeja.

D3-kirjastoa tullaan todennäköisesti kehittämään vielä pitkän aikaa ja käyttö todennäköisesti lisääntyy tulevaisuudessa varsinkin, jos D3-kirjastolle sopivien visualisointien suosio kasvaa.

6.3 Explorable explanations -käsite

Yksi hyvin uusi ja vielä melko tuntematon käsite on explorable explanations, joka suoraan suomennettuna tarkoittaa tutkittavat selitykset. Tällä termillä tarkoitetaan jonkin asian opettelemista itse kokeilemalla animoitujen kuvioiden ja kuvaajien ja muiden vastaavien visuaalisten apuvälineiden avulla, jotka kannustavat tekemään ja tutkimaan.

Yksi hyvä esimerkki explorable explanation:sta on Maartin Lambrecht:n tekemä Rock ’n Poll. Hän haluaa opettaa tällä, että mielipidetutkimuksissa tulee eroja, vaikka se toistettaisiin saman populaation sisällä ja että harvoin tulokset vastaavat täydellisesti todellisuutta.



KUVA 11. Mielipidetutkimusten tuloksien tutkimista (Rock ’n Poll 2016)

Kuvassa 11 on alemmassa palkissa oikea jakauma ja ylempänä jakauma, kun asiaa on kysytty tuhannelta ihmiseltä. Myöhemmissä vaiheissa sovellusta näitä kyselyjä voidaan toteuttaa useita samanaikaisesti ja nähdään helposti, että erot vaihtelevat kyselyittäin ja välillä virhemarginaali saattaa olla useita prosentteja.

7 POHDINTA

Datavisualisointi itsessään on vanha käsite, mutta moderni tapa visualisoida dataa on erilainen kuin vaikkapa kymmenen vuotta sitten. Internetin ja mobiililaitteiden suosion myötä vaatimukset hyville, responsiivisille ja interaktiivisille visualisoinneille erityisesti mediassa on kasvanut huimasti. Yksinkertaisia kuvaajia voi edelleen tietenkin tehdä yksinkertaisilla työkaluilla, mutta suuren datamäärän visualisointi käyttäjäystävällisellä tavalla vaatii kuitenkin parempia työkaluja ja enemmän osaamista. The New York Times onkin julkaissut viime vuosina laajoja interaktiivisia visualisointeja monista tärkeistä muun muassa yhteiskunnallisista asioista ja ne ovat olleet todella suosittuja.

Työkaluja datan visualisoimiseen on paljon, mutta useimmat kehittyneimmistä työkaluista ovat tarkoitettu nopeaan visualisointiin, jossa ei itse saa juurikaan vaikuttaa graafin yksityiskohtiin. D3-kirjasto on poikkeus tähän ja sen takia se pystyykin kilpailemaan esimerkiksi Google Charts:n kanssa. D3-kirjaston suosion voisi kuvitella nousevan tulevaisuudessa erityisesti sellaisten palvelujen tekemisessä, jotka vaativat paljon monipuolista interaktiivisuutta monimutkaiselle datalle.

Työn aikana tehtyä projektia ei jatketa, mutta sen aikana opittuja asioita käytetään hyväksi muissa projekteissa. Työssä onnistui erityisen hyvin datan hallinta siten, että kuvaajasta pystyi helposti esimerkiksi piilottamaan tietyn vastausvaihtoehdon pylvään tai jakamaan pylväät pienempiin osiin vastaajien taustatietojen perusteella. Työssä parantamisen varaa jäi koodin optimointiin siten, että se ei turhaan tuhlaisi laitteen resursseja esimerkiksi piirtämällä näkyvissä olevia asioita uudestaan silloin, kun ei ole tarvetta.

Työssä oppi paljon D3-kirjaston käyttämisestä. Erityisesti aikaisempi kokemus JavaScript:stä ja jQuery-kirjastosta helpotti oppimista ja saivat D3-kirjaston syntaksin ja toiminnan tuntumaan selkeiltä heti alusta asti. Hyvin yksinkertaisen peruskuvaajan tekeminen ei vie enää muutamaa minuuttia kauempaa. Vaikeampiin kuvaajiin menee kuitenkin kauan aikaa, esimerkiksi kun data on monitasoista ja kuvaajaan haluaa saada kaiken näkyviin järkevästi. Kirjaston dokumentointi on kuitenkin erittäin hyvin toteutettu, joten kaikki tarvittava tieto on löydettävissä.

LÄHTEET

Shamas, N. 2015. Why data visualization is important. Luettu 20.10.2016.
https://www.techchange.org/2015/05/19/data-visualization-analysis-international-development/

OCSI. 2009. Case study: What is good visualisation? Luettu 13.11.2016.
http://www.improving-visualisation.org/case-studies/id=6

Thompson, C. 2016. The Surprising History of the Infographic. Luettu 20.10.2016.
http://www.smithsonianmag.com/history/surprising-history-infographic-180959563/

Bostock, M. 2012. For Protovis Users. Luettu 13.11.2016.
http://mbostock.github.io/d3/tutorial/protovis.html

CSS3 .info. 2012. CSS3 Module Status. Luettu 25.10.2016.
http://www.css3.info/modules/

W3Techs. 2016. Usage of JavaScript libraries for websites. Luettu 27.10.2016.
https://w3techs.com/technologies/overview/javascript_library/all

Lambrechts, M. 2016. Rock ‘n Poll. Luettu 9.11.2016.
http://rocknpoll.graphics/

Bostock, M. 2016. Open questions #47. Luettu 9.11.2016
https://github.com/d3/d3-selection/issues/47#issuecomment-173413922

Bostock, M. Carter, S. Tse, A. 2014. Is It Better to Rent or Buy? Luettu 16.11.2016.
http://www.nytimes.com/interactive/2014/upshot/buy-rent-calculator.html

LIITTEET

Liite 1. Projektin rakenne