OULUN AMMATTIKORKEAKOULU

Janne Paakkari

PÄIVÄKIRJAMALLINEN OPINNÄYTETYÖ FRONT-END-OHJELMISTOKEHITTÄJÄN NÄKÖKULMASTA

PÄIVÄKIRJAMALLINEN OPINNÄYTETYÖ FRONT-END-OHJELMISTOKEHITTÄJÄN NÄKÖKULMASTA

Janne Paakkari
Opinnäytetyö
Kesä 2018
Tietojenkäsittelyn tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietojenkäsittelyn tutkinto-ohjelma, Web-sovelluskehityksen suuntautumisvaihtoehto

---

Tekijä(t): Janne Paakkari
Opinnäytetyön nimi: Päiväkirjamallinen opinnäytetyö front-end-ohjelmistokehittäjän näkökulmasta
Työn ohjaaja: Ritva Virkkala
Työn valmistumislukukausi ja -vuosi: Syksy 2018 Sivumäärä: 55

Tässä portfoliomaisessa päiväkirjamuotoisessa opinnäytetyössä seurataan opiskelijan työtehtäviä 10 viikon ajan yrityksessä, joka sijaitsee Oulussa. Opinnäytetyössä kirjoitetaan päiväkirjaa päivittäin ja viikon lopuksi tehdään analyysi perustuen viikon aikana ilmenneisiin asioihin. Opiskelijalla on voimassa oleva salassapitosopimus, joten se on täytynyt ottaa huomioon opinnäytetyötä kirjoittaessa. Työtehtävät ovat luonteeltaan front-end ohjelmistokehitystä.

Tietoperustana on käytetty virallisten dokumentaatioiden ohella erilaisia blogeja, sekä artikkeleita. Ohjelmointikielet ja ohjelmointikäytänteet muuttuvat varsin nopeasti ja uusimman tiedon sekä uusimmat vinkit saa yleensä vertaisilta netin välityksellä.

Työn tavoitteena oli luoda ikkuna front-end ohjelmistokehityksestä kiinnostuneille henkilöille, jotka voivat opinnäytetyötä lukiessa tutustua tyypillisiin tehtäviin ja haasteisiin, joita varsinkin aloitteleva front-end ohjelmistokehittäjä voi kohdata. Toisaalta myös työnantajilla ja muilla aiheesta kiinnostuneilla on mahdollisuus nähdä polku mitä aloittelevat ohjelmistokehittäjät saattavat kulkea ja sitä kautta muokata esimerkiksi omia käytänteitä parhaiden mahdollisten tulosten saamiseksi uusien rekrytoitavien kanssa.

Opinnäytetyötä voidaan siis hyödyntää jatkossa käyttämällä sitä esimerkiksi organisaation kehityksen tukena, tai alasta kiinnostuneen henkilön ammatinvalinnan selkeyttämisessä.

Asiasanat: Ohjelmistokehitys, ReactJS, etätyö, päiväkirja

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Business Information Systems, Option of Web Application Development

---

Author(s): Janne Paakkari
Title of thesis: A diary-based thesis from a front-end software developer's point of view
Supervisor(s): Ritva Virkkala
Term and year when the thesis was submitted: Fall 2018 Number of pages: 55

The topic of this portfolio type of diary-based thesis was to follow the author`s job assignments for ten weeks on company located at Oulu. Author wrote a diary on daily basis and at the end of the week author wrote a weekly analysis which was based on issues which occurred during the week. The author had a valid confidentiality agreement, so he must have taken it into account when writing the thesis. The job assignments dealt with front-end software development.

The theoretical background consisted of official documentations, blogs and articles. Programming practices tend to change very often; therefore, the latest information is usually acquired on a peer-to-peer basis. Compared to online sources, traditional books cannot usually keep up with the most recent changes on world of programming.

The aim of the thesis was to create a window for people interested in front-end software development. Those people coming in similar background with the student can read about the typical challenges a front-end software developer may face. People of similar background may then enhance their career choice. In addition, employers and other people interested in the topic also have an opportunity to see what challenges a new software developer may face and then for example improve their existing policies of the company.

Finally, the author achieved diary-based thesis in accordance with the objectives. This thesis can be utilized in the future for someone professionally interested in front-end-developing by enhancing their view of their career choice or by organizations by providing ideas how to develop the organization further.

Keywords: Software development, ReactJS, remote work, diary

SISÄLLYS

1 JOHDANTO

Opinnäytetyöni päiväkirjaosuus on kirjoitettu 7.5.2018–13.7.2018 välisenä aikana. Opinnäytetyöni kattaa siis 10 viikkoa. Kirjoitin päiväraportin joka päivä, sekä viikon päätteeksi erillisen viikkoanalyysin.

Päiväraporteissa on lyhyt työtehtävien kuvaus kyseiseltä päivältä sekä omat tavoitteeni ja analyysi siitä, miten olen onnistunut päivittäisissä tehtävissäni. Viikkoanalyyseissa on analyysi perustuen päiväraportteihin, kirjallisuuteen, käytössä oleviin toimintamalleihini, viikon aikana mieleen tulleihin asioihin, sekä kohtaamiini ongelmiin.

Työskentelen Oulussa sijaitsevassa yrityksessä front-end-ohjelmistokehityksen parissa. Yritys, jossa työskentelen, on perustettu vuonna 2018, joten sitä voisi kuvailla startupiksi. Yritys tarjoaa erilaisia ohjelmiston suunnittelu- ja valmistus palveluja. Työskentelen yrityksessä pääsääntöisesti etänä 8t päivässä sisältäen puolen tunnin ruokatauon. Varsinaista työaikaa minulla ei ole. Pidän vain huolta siitä, että sovitut tunnit tulevat tehtyä. Käytännössä työskentelen kuitenkin yleensä klo 8-16 välisenä aikana johtuen siitä, että silloin on helpompi olla yhteydessä muihin yrityksessä työskenteleviin henkilöihin.

Työtehtäväni vaativat varsinaisten ohjelmointitaitojen lisäksi kokemusta versionhallinnasta, palaverikäytännöistä sekä asiakasrajapinnassa toimimisesta. Lisäksi tiedonhakuun liittyvät taidot ovat välttämättömiä ja, koska materiaalit ovat yleensä englanniksi, niin myös englantia on pakko ymmärtää.

Aloitin työni samaan aikaan kuin aloitin opinnäytetyöni kirjoittamisen. Opinnäytetyöni tarjoaakin siksi erinomaisen mahdollisuuden seurata aloittavan ohjelmistokehittäjän arkea, jonka tuloksia voidaan käyttää esimerkiksi analysoidessa opiskelijoiden tosiasiallista osaamistasoa verrattuna yrityselämän vaatimuksiin, vaikka kyseessä ovatkin tietysti vain yhden opiskelijan kokemukset. Osasin töissäni vaadituista asioista käytännössä sen, minkä tarvitsinkin alkuun pääsemiseen. Ammattikorkeakoulussani on alusta asti opiskeltu varsin paljon käytännön kautta, mikä auttoi alkuun pääsemisessä. Eniten käyttämäni tekniikka oli ReactJS, joka on JavaScript-kirjasto. En osannut alussa kuin perusteet, mutta se ei haitannut työtehtävissäni suoriutumista, sillä omasin muista kielistä hyvän pohjan, jonka ansioista opin myös ReactJS:n perusteet varsin nopeasti.

Yleisenä heikkoutenani oli alussa ehkä teorian osaamisen puute. Esimerkiksi JavaScriptin eri versiot aiheuttivat aluksi hieman hämmennystä, vaikka asiaa oli ammattikorkeakoulussakin käyty läpi. Työtehtävistä suoriutumisen osalta tältä osin ei kuitenkaan suurempia ongelmia ilmennyt.

2 NYKYTILANTEEN KUVAUS

2.1 Oman nykyisen työn analyysi

Aloitin työt samaan aikaan opinnäytetyöni kanssa, joten alkutilanne oli sen mukainen. Työtehtäväni sisältävät kuitenkin front-end-ohjelmointia ja siihen liittyvää oheistoimintaa, kuten versionhallinnan käyttöä sekä asiakasrajapinnassa toimimista. Olen esimerkiksi säännöllisesti erilaisissa palavereissa asiakkaiden kanssa. Tyypillinen päivä voisi tehtävien osalta esimerkiksi olla seuraavanlainen:

- Sähköpostin tarkastus ja sähköposteihin vastaaminen
- Annettujen ohjelmointispeksien tarkasteleminen
- Tiedonhakua ja suunnittelua tehtävän toteuttamiseen
- Ohjelmointia
- Palaveri
- Viimeistelyä tekemääni koodiin
- Koodin versionhallintaan toimitus.

Työssäni tarvitaan osaamista, joka liittyy ohjelmistoympäristöihin, versionhallintaan, itse ohjelmointiin ja asiakkaiden kanssa toimimiseen. Käytännössä myös englannin kielen osaaminen on välttämätöntä, mikä johtuu suomalaisten materiaalien vähäisyydestä ja vanhentuneisuudesta. Perustason osaaminen riittää edellä mainittuihin asioihin, sillä työtehtävissä osaaminen kehittyy.

Tarvittavista alkutiedoista tärkein on tieto siitä, miten itse ohjelmointia käytännössä toteutetaan. Täytyy siis tietää, miksi ohjelmoinnissa käytetään eri ohjelmointikieliä ja miten niiden käytössä pääsee alkuun. Osaamisen ja tiedon ero tulee esille siinä, että tarvittavia kieliä ei tarvitse osata heti, mutta tieto esimerkiksi yleisesti käytettävistä lauseista, kuten ehtolauseista, on välttämätöntä. Olisi hyvä tietää myös perusteet käyttöliittymään ja käyttäjäkokemukseen vaikuttavista asioista, mainittakoon hahmolait. Tärkeintä kuitenkin on tietää, mistä tietoa voi hakea lisää. Jos eteen tulee esimerkiksi uusi kieli/kirjasto/viitekehys, niin on hyvä tietää, että yleensä kaikilla on jonkinlaiset dokumentaatiot, joilla pääsee vähintäänkin alkuun.

Välttämättömiä taitoja ovat erityisesti ongelmanratkaisutaidot, ihmissuhdetaidot ja tiedonhankintataidot. Vaikka olisi täydelliset tiedot ja mekaaninen osaaminen ohjelmointiin liittyen, niin välillä voi joutua ratkaisemaan ongelmia, jotka johtuvatkin ulkopuolisesta asiasta, kuten käytössä olevasta tietokoneesta ja sen konfiguroinnista. Tällöin ongelma on hyvä huomata, rajata ja ratkaista mahdollisimman aikaisessa vaiheessa. Myös itse ohjelmointikehitys on pohjimmiltaan monesti aika luovaa ongelmanratkaisua luonteeltaan. Ihmissuhdetaidot tulevat taas esille siinä, että ohjelmistokehitys on tiimityöskentelyä, mistä kertoo esimerkiksi se, että palavereita on tyypillisesti useampikin viikkoa kohti. Palavareiden ulkopuolella ollaan lisäksi yhteydessä yksittäisiin henkilöihin lähes päivittäin. Uutta tietoa tulee säännöllisesti lisää, ja täytyy osata myös itse kertoa missä tilanteessa mennään. Tiedonhankintataidot taas ovat sen takia välttämättömiä, että ilman niitä on vaikea kehittyä ja löytää optimaalisia ratkaisuja ongelmiin. Vaikka ongelmanratkaisutaidot olisivat huippuluokkaa ja ratkaisisit tietyn ongelman itse, niin todennäköisesti se ei kuitenkaan olisi paras tai välttämättä edes hyvä ratkaisu. Tällöin siis olisi hyvä osata etsiä muiden ratkaisuja ongelmiin ja verrata omaa ratkaisua suhteessa muihin ratkaisuihin.

Koska oma tilanteeni on se, että olen tullut töihin suoraan ammattikorkeakoulusta, niin aikaisempi osaamiseni on pääsääntöisesti peräisin myös sieltä. Arvioisin siitä huolimatta, että osaamiseni on suhteessa työtehtävien osaamisvaatimuksiin taitavan suoriutujan tasolla. Taitava suoriutuja tarkoittaa tässä yhteydessä sitä, että kykenen suoriutumaan tehtävistäni itsenäisesti. Ammattikorkeakoulussa sain hyvät valmiudet tiedon hankkimiseen ja osaankin yleensä hankkia relevanttia tietoa minulle annettuihin tehtäviini. Vinkit ohjelmointikäytäntöjeni parantamiseksi ovat kuitenkin tervetulleita, sillä en kuitenkaan vielä ole kokeneen asiantuntijan tasolla. Kokenut asiantuntija tarkoittaisi sitä, että kykenisin ohjaamaan ja opastamaan muiden toimintaa ja kehittämään työtehtävässä vaadittavia toimintamalleja.

Ammatillisessa kehittymisessäni olen tasolla edistynyt, vaikka minulla on parannettavaa hieman jokaisella osa-alueella. Hallitsen kuitenkin ohjelmoinnista perusteet, mikä näkyy siinä, että kykenen suoriutumaan itsenäisesti yleensä lähes jokaisesta tehtävästä. Olen myös ohjelmoinut lähes kaksi vuotta niin ammattikorkeakoulussa kuin hieman vapaa-ajallanikin. Olen käyttänyt ohjelmissani useaa eri ohjelmointikieltä ja erilaisia tekniikoita, joten en kutsuisi itseäni enää tässä vaiheessa aloittelijaksi vaan edistyneeksi, kuten jo alussa mainitsinkin. Kysymys onkin tässä vaiheessa siitä, että kauan minulla menee aikaa tietyn tehtävän ratkaisemiseen ja kuinka optimoitu mahdollinen ratkaisuni tosiasiallisesti on. Mielestäni järkevin asia mitä, tässä vaiheessa voin

tehdä ammattitaitoni kehittymisen kannalta, on yksinkertaisesti tehdä alan työtä ja saada parempaa rutiinia sitä kautta. Minun tulisi enemmän pyrkiä noudattamaan alan käytänteitä, olen välillä esimerkiksi ammattikorkeakoulussa pyrkinyt vain saamaan ohjelman toimimaan, joka ei luonnollisesti ole mikään paras käytäntö. Pelkkä mekaaninen toimivuus ei riitä vaan täytyy pyrkiä tekemään koodia, jota voi myös järkevästi ylläpitää ja perustella, esimerkiksi tietoturvan näkökulmasta. Pyrin kuitenkin koko ajan eteenpäin ja seuraamaan alalla tapahtuvia muutoksia ja kehittymään ammatillisesti mahdollisimman hyväksi.

2.2 Sidosryhmät työpaikalla

Minun näkökulmastani katsottuna työpaikkani sisäiset sidosryhmät käsittävät esimiehet, muut ohjelmoijat ja graafikot. Ulkoiset sidosryhmät sisältävät asiakkaat ja yhteistyökumppanit. Todennäköisesti muitakin sidosryhmiä, kuten viranomaisia, kuuluu yrityksen sidosryhmiin, mutta ne eivät kosketa työtäni suoranaisesti, joten en ole laskenut niitä joukkoon. Sidosryhmät on kuvattu myös kuviossa yksi paremman graafisen kuvan saamiseksi.

KUVIO 1. Sidosryhmät

Esimiehet päättävät, mitä teen ja mitä en tee. Minun täytyy luonnollisesti myös kommunikoida esimiesten kanssa, miten olen edennyt ja miten tulen jatkossa etenemään. Muiden ohjelmoijien kanssa minun pitää vähintäänkin varmistaa se, että emme tee mitään asioita päällekkäin ja saada yleinen tilannekuva siitä, mitä kukakin on tekemässä. Mahdollisissa ongelmatilanteissa muiden

työntekijöiden apu on myös korvaamatonta, ja vastaavanlaisesti minun täytyy olla valmiina auttamaan myös muita mahdollisten ongelmien yllättäessä. Graafikot tuottavat nimensä mukaisesti grafiikkaa, jota toisinaan yhteensovitan töissäni, joten olen myös yhteydessä heihin. Yhteistyökumppanit sekä asiakkaat ovat luonnollisesti tärkeitä yrityksen toiminnan kannalta, mutta tässä en voi mennä kovin syvälle, sillä salassapitosopimus koskee erityisesti juuri tätä osa-aluetta.

2.3 Vuorovaikutustaidot työpaikalla

Teen paljon etätöitä, minkä vuoksi yhteydenpidon merkitys korostuu verrattuna siihen, että kaikki työskentelisivät toimistolla. Pääsääntöisesti pidän yhteyttä niin työkavereihin, kuin asiakkaisiinkin sähköisesti, puhelimen välityksellä tai erilaisten palaverien kautta kasvotusten. Työkavereiden kanssa tulee lähinnä jaettua erilaista tietoa ja kysyttyä asioita, sekä tehtyä erilaisia tarkennuksia työhön liittyen. Asiakkaiden kanssa kommunikoidessa on tärkeää osata kommunikoida asianmukaisesti ja säntillisesti, jottei epäselvyyksiä tule. Käydyistä asioista olisi hyvä laatia muistio. Erona vuorovaikutukseen työkavereiden kanssa on se, että vuorovaikutus asiakkaiden kanssa ei ole yhtä paljon teknillisyyteen painottuvaa (=miten tehdään), vaan kysymyksinä on yleensä, mitä tehdään ja milloin tehdään.

Sähköisten kanavien kautta vuorovaikutus tapahtuu luonnollisesti etänä, mikä luo omat haasteensa, sillä toista ihmistä ei näe suoraan, jolloin esimerkiksi väärinymmärrysten riski voi kasvaa. Etuna esimerkiksi sähköposteilla kommunikoidessa on kuitenkin se, että sovitut asiat voidaan aina tarkastaa jälkikäteen ja ne eivät unohdu niin helposti, kun on myös jotain kirjallista.

Puhelimella soittaessa suurin haaste on yleensä siinä, että kynää tai paperia ei välttämättä ole saatavilla tai on muuten huono hetki, sillä keskittyminen saattaa olla esimerkiksi jonkin ohjelmointiin liittyvän ongelman ratkaisemissa. Soittaessa asiat eivät muutenkaan mielestäni tule yleensä yhtä selviksi kuin suullisesti tai sähköisesti kommunikoimalla. Puhelin on kuitenkin kätevä välinne lyhyiden tilanneraporttien antoon tai pienien asioiden selvittämiseen.

Suullisesti kommunikoidessa haasteet liittyvät lähinnä muistiinpanojen tekemiseen. On tärkeää osata kirjata oman työn kannalta olennaisimmat asiat ylös siten, että niitä voi myöhemmin käyttää muistin tukena. Pelkkään muistiin luottaessa ainakin itselläni unohtuu varsin merkittävä osa sovituista asioista.

Yleisesti ottaen niin suullisesti, soittaen, kuin sähköisesti kommunikoidessa täytyy ottaa huomioon keskustelukumppanin vahvuudet ja heikkoudet. Esimerkiksi liian teknisistä asioista ei kannata puhua, jos ne eivät kuulu keskustelukumppanin vahvuusalueisiin, vaan tällöin kannattaa löytää jokin keskitie mitä pitkin kulkea. Omalta osalta täytyy myös tiedostaa omat vahvuudet ja heikkoudet ja reilusti sanoa, jos itse ei ymmärrä kunnolla, mistä keskustelukumppani puhuu.

3 PÄIVÄKIRJA

3.1 VIIKKO 19

Maanantai 7.5.2018

Päivän tavoitteena oli yksinkertaisuudessaan saada vain hyvä yleiskuva tulevista työtehtävistäni ja siitä, miten niitä käytännössä tehdään. En tarkemmin tiennyt, mitä tuleman pitää. Tiesin ainoastaan sen, että olen menossa tekemään ohjelmistokehitystä sekä suunnittelua painottuen todennäköisesti kehitykseen.

Päivä oli kokonaisuudessaan aika pitkälti odotusteni mukainen. Käytännössä työni on aluksi front-end-ohjelmistokehitystä. Kävimme läpi yleisiä käytänteitä ja työtehtäviä, lisäksi minua perehdytettiin työtehtäviini esimiesteni toimesta. Puhuimme myös salassapitosopimuksesta ja siitä, mistä tässä päiväkirjassa voi kirjoittaa. Salassapitosopimuksen takia joudun kirjoittamaan töistäni ja tekemisistäni aika yleisluonteisesti. Fokus päiväkirjassani onkin siinä, miten tietojenkäsittelyn tutkinto-ohjelma oikeasti valmistaa alan työtehtäviin.

Asensin myös tarvittavan ohjelmointiympäristön kannettavaani sekä päivän jälkeen myös pöytäkoneeseeni. Olen osan työajasta etätöissä, joten on tärkeää, että molemmissa koneissa ohjelmointiympäristöt ovat kunnossa. Työssäni käyttämäni ohjelmistoympäristö vastaa ensimmäisen päivän käsitykseni mukaan aika pitkälti sitä, mitä olen käyttänyt ammattikorkeakoulussa. Eroavaisuudeksi sanoisin sen, että ammattikorkeakoulun vastaavat ympäristöt ovat yleensä olleet enemmän graafisesti säädettävissä. Esimerkkinä mainittakoon Netbeans, jossa komentoriviä ei käytännössä tarvitse käyttää ollenkaan.

Tiistai 8.5.2018

Päivän tavoitteena olisi päästä hiljalleen itse ohjelmointiin käsiksi, jotta käytettävät teknologiat tulisivat enemmän tutuksi. Tarkemmista tehtävistä en oikeastaan päivän aluksi tiennyt.

Päivän aluksi meillä oli palaveri, jossa selvisi se, mitä lähden ohjelmoimaan ja miten. Pääteknologiana on ReactJS, jota kokemusteni mukaan käytetäänkin aika laajasti eri sovelluksissa. Olen myös hieman käyttänyt ReactJS:ää muutama kuukausi sitten. Minulla on luonnollisesti kuitenkin siitä vielä paljon opittavaa, ja sen käyttö on saattanut osittain unohtuakin.

Tutustuimme palaverin jälkeen siihen, miten ReactJS toimii käytännössä. Katsoin myös vanhoja ammattikorkeakoulussa tekemiäni töitäni muistin virkistykseksi. Olen ohjelmoinut esimerkiksi ristinollaa ja laivanupotusta ReactJS:ää apuna käyttäen, vaikka ne ovatkin suhteellisen alkeellisia ohjelmia. Loppupäivästä tutustuin ReactJS:ään itsekseni ja aloitin myös ensimmäiset työtehtäväni, jotka sisälsivät enimmäkseen tiedon hankintaa ja soveltamista.

Olen tyytyväinen päivään, sillä sain myös seuraavalle päivälle jotain näytettävää ja tiedonhankintani tuotti tulosta. Lisäksi ReactJS alkoi pikkuhiljaa tulemaan takaisin mieleeni. ReactJS:n syntaksi on sinänsä looginen. Se kuitenkin eroaa aika paljon esimerkiksi puhtaasta JavaScriptistä, jonka tyyppisiin kieliin olen tutustunut enemmän.

Keskiviikko 9.5.2018

Tänään päivän tavoitteena on saada aikaan joitain suurempia ohjelmapalasia sekä oppia ReactJS:tä taas jotain uutta. Minulla on aamusta myös palaveri, jossa suunnittelemme hieman, mitä minun kannattaisi huomiseksi taas tehdä.

Käytännössä rupesin alustamaan tulevaa projektiani. Päivän opiksi jäi oikeastaan se, mitä frameworkeja ja kirjastoja ReactJS:n kanssa voi oikeasti käyttää. Esimerkiksi se tuli hieman yllätyksenä, että puhdasta Bootstrap-frameworkia ei voi käyttää kovinkaan hyvin sellaisenaan. Jos haluaa käyttää Bootstrappia vastaavaa frameworkkia, täytyy käyttää React-Bootstrapia tai Reactstrapia. Bootstrapin käyttö ei ole kuitenkaan ainoa esimerkki, johon törmäsin, mutta se jäi eniten mieleen, sillä ammattikorkeakoulussa olen käyttänyt Bootstrapia lähes jokaisessa projektissani, jos se on ollut mahdollista. Olen tyytyväinen päivään, sillä olen palauttanut ReactJS:n jo aika hyvin mieleeni, sekä oppinut myös hieman uutta.

Torstai 10.5.2018

Tavoitteekseni asetan jälleen uuden oppimisen ja erityisesti sen, miten ReactJS:n rakennetta saadaan optimoitua mahdollisimman hyvin. Se tarkoittaa sitä, että suunnittelen mitä, miten ja milloin ohjelmoida mitäkin. Minulla on suhteellisen vapaat kädet ohjelmointiin. Näytän tuotokseni säännöllisin väliajoin ja saan siitä palautetta, jonka perusteella pyrin kehittämään ohjelmointitaitojani.

Sain tänään aikaiseksi ensimmäisiä versioita tekemistäni töistä versiohallintaan asti. Tein oman branchin, mikä oli minulle uutta. Olen yleensä laittanut tekemäni ammattikorkeakoulutyöt suoraan vain master-linjaan eli päälinjaan. Tämä toimii, koska ammattikorkeakoulussa työt ovat pääsääntöisesti olleet yksilötöitä ja ryhmätöissä olemme käyttäneet pilvialustaa (Cloud9), jossa muutokset näkyvät heti kaikille ja versio on aina ajan tasalla. Olisin kuitenkin voinut hieman harjoitella tätä jo ammattikorkeakoulussa, sillä tiesin kyllä jo silloin, että työelämässä harvemmin käytetään pelkkää master branchia.

Mielestäni suunnittelin melko hyvin sen, mitä teen seuraavina päivinä. Hahmottelin paperille, mitä tarvitaan ja missä järjestyksessä. Tein vastaavat kohdat ohjelmaan ja tein tärkeimmille kohdille jo hieman pohjaa. Tästä on hyvä jatkaa.

Perjantai 11.5.2018

Tänään oli etäpäivä eikä palavereita ollut. Päivän tavoitteena oli viimeistellä nettisivut WordPressiä käyttäen.

WordPress-alusta on ammattikorkeakoulusta tuttu. Olen tehnyt nettisivuja niin omalla teemalla, kuin valmisteemallakin. WordPress-sivujen tekeminen on aluksi aina pääsääntöisesti konfigurointia. Esimerkiksi php.ini-tiedostosta on hyvä käydä muuttamassa tiedoston lähetysten maksimikoko hieman isommaksi, sillä oletuksena se on vain 2 MB.

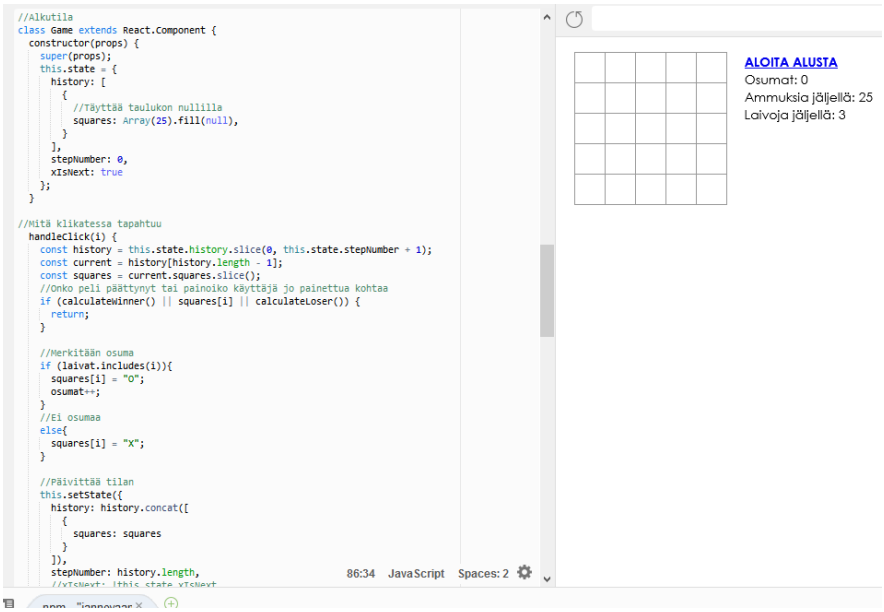
Tänään teema ja käyttämäni lisäosat olivat hieman tuntemattomia minulle, joten niiden opettelemiseen meni aikaa. Ne eivät olleet vaikeita, vaan aika meni lähinnä UI:n hahmottamiseen eli siihen, mitä on missäkin. Siksi en täysin päässyt tavoitteeseeni. Sain sivuille rungon valmiiksi, mutta asettelut ja muut jäivät puolitiehen.

Viikkoanalyysi

Viikko kului melko nopeasti. Alkuviikko meni pääsääntöisesti ympäristöjen ja teknologioiden ymmärtämiseen. Pääsin kuitenkin aika nopeasti kiinni työn kuvaani.

Jouduin selvittelemään aika paljon asioita dokumentaatioista sekä ammattikorkeakoulussa tehdyistä töistäni. Katsoin esimerkiksi ReactJS:n perusasiat laivanupotuspelistäni, jonka tein noin muutama kuukausi sitten ammattikorkeakoulussa. Kyseinen laivanupotuspeli on aika alkeellinen, ja siinä on hieman myös niin sanottua kuollutta koodia mukana, sillä se perustui aikaisemmin tehtyyn ristinollaan. Kuollut koodi ei ole koskaan hyvä asia, sillä se vaikeuttaa koodin ymmärtämistä tai saattaa aiheuttaa bugeja. Bugi voi esimerkiksi aiheutua siitä, jos kuollut koodi herääkin yhtäkkiä henkiin ja alkaa sotkemaan ohjelman toimintaa. Kuollut koodi pitää siis yksinkertaisesti vain poistaa. Sen löytäminen ei kuitenkaan aina ole niin helppoa kuin kuvittelisi, ja jopa isoilla firmoilla saattaa tulla ongelmia sen kanssa. (InfoQ 2017, viitattu 12.5.2018).

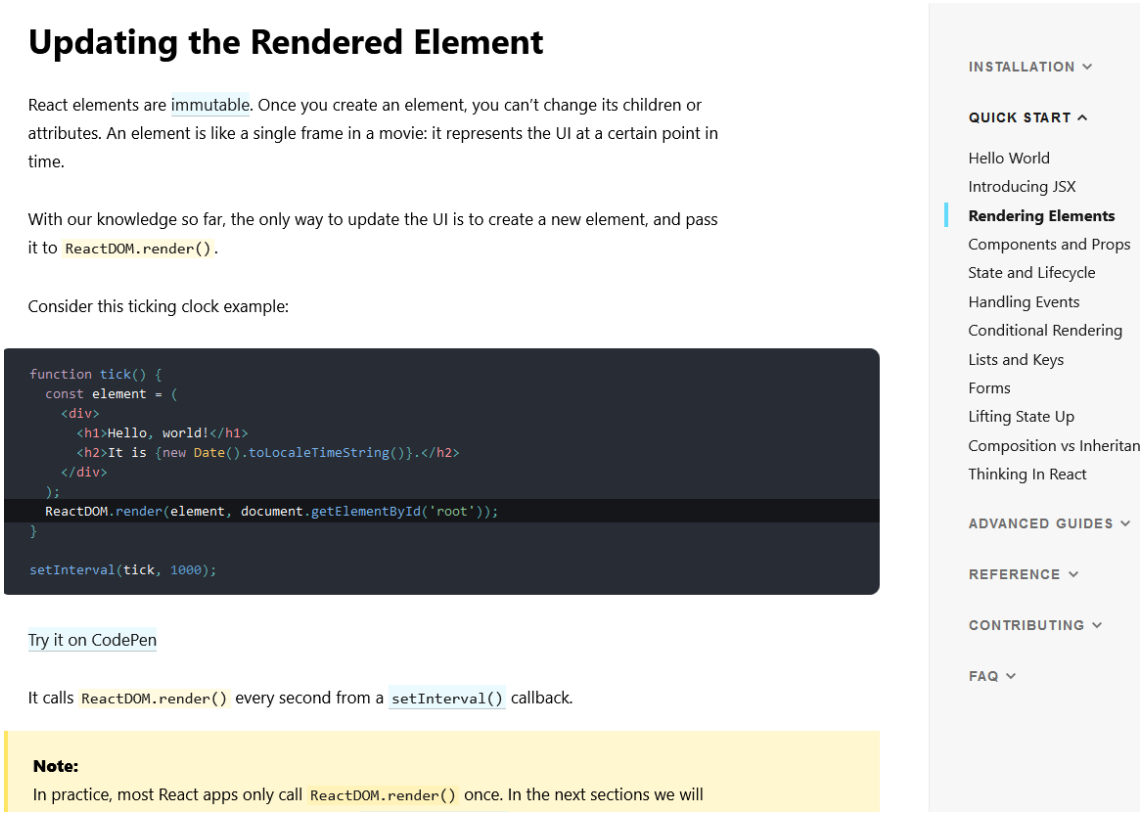
Olin kuitenkin kommentoinut laivanupotuspelini suhteellisen hyvin, vaikkei se mikään selitys kuolleen koodin olemassaololle olekaan. Kommentit auttoivat kuitenkin ymmärtämään ja muistamaan nopeasti, mitä olin ajatellut peliäni tehdessä.



KUVIO 2. Näyte ammattikorkeakoulussa tekemästäni laivanupotuspelistä

Töissäni pyrin luonnollisesti välttämään kuolleen koodin kirjoittamista, eikä toistaiseksi sen kanssa ole ollut ongelmia.

Dokumentaatiot olivat ammattikorkeakoulutöideni lisäksi toinen asia, jota katsoin aika paljon. Dokumentaatiot ovat yleensä onneksi aika selkeitä ja ymmärrettäviä. ReactJS:n dokumentaatio perustuu pitkälti esimerkkeihin ja onkin siten enemmän tutoriaalityyppinen, kuin lista asioita (ReactJS 2018, viitattu 12.5.2018).



Updating the Rendered Element

React elements are immutable. Once you create an element, you can't change its children or attributes. An element is like a single frame in a movie: it represents the UI at a certain point in time.

With our knowledge so far, the only way to update the UI is to create a new element, and pass it to `ReactDOM.render()`.

Consider this ticking clock example:

```
function tick() {
  const element = (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
  );
  ReactDOM.render(element, document.getElementById('root'));
}

setInterval(tick, 1000);
```

Try it on CodePen

It calls `ReactDOM.render()` every second from a `setInterval()` callback.

Note:
In practice, most React apps only call `ReactDOM.render()` once. In the next sections we will

INSTALLATION
QUICK START
Hello World
Introducing JSX
Rendering Elements
Components and Props
State and Lifecycle
Handling Events
Conditional Rendering
Lists and Keys
Forms
Lifting State Up
Composition vs Inheritan
Thinking In React
ADVANCED GUIDES
REFERENCE
CONTRIBUTING
FAQ

KUVIO 3. Yksinkertainen esimerkki ReactJS:n dokumentaatiosta

Viikon viimeisen päivän eli perjantain selvisin pitkälti omilla tiedoillani, eikä minun tarvinnut katsoa WordPressin omia dokumentaatiota. Jouduin kuitenkin katsomaan muutaman käytössä olevan lisäosan tutoriaaleja. En voi tarkemmin avata lisäosa- tai teemastackia salassapitosopimuksen takia.

Tällä viikolla tein myös aika paljon etätöitä, kuten teen näillä näkymin myös tulevaisuudessa. Etätöissä kommunikaatio korostuu, ja se korostuu vielä sitäkin enemmän näin uuden työn alussa. Olin kuitenkin perjantaita lukuun ottamatta palavereiden kautta yhteydessä työni kannalta relevantteihin henkilöihin. Perjantaina hoidin kontaktit sähköpostin avulla.

3.2 VIIKKO 20

Maanantai 14.5.2018

Tälle päivälle ei ollut oikeastaan muita tavoitteita kuin saada viikko hyvään vauhtiin. Normaalisti pyrin päivittäin asettamaan jonkin hieman konkreettisemman tavoitteen, mutta nyt tein poikkeuksen sääntöön. Tänään aamulla oli taas palaveri. Katsoimme tilannetta, missä olemme ja mitä pitäisi saada seuraavaksi aikaiseksi. Tämän lisäksi tiedossa oli nettisivujen tekemistä sekä muuta front-end-ohjelmointia.

Tein aamun palaverin jälkeen tyypillistä front-end-ohjelmointia ReactJS:ää käyttäen. Sain tehtyä uusia ominaisuuksia, jotka näyttivät toimivan ainakin eräänlaisessa rautalankamallissa. Ohjelmointityylinäni on siis hieman kokeilla asiaa x tai y ylimalkaisesti ja katsoa, onko siinä potentiaalia. Erittäin tutut asiat pystyn yleensä aika pitkälti suunnittelemaan, eikä mitään kokeiluja tarvitse silloin tehdä. Nettisivuihin liittyen mitään dramaattisempaa ei tapahtunut suuntaan tai toiseen. Mitään suurempaa en saanut aikaiseksi, vaan sain lähinnä paranneltua aikaisempaa rakennetta ja ratkaistua muutamia pienempiä ongelmia. Tavoitteen voisi sanoa täyttyneen, sillä viikko lähti kuitenkin hyvin käyntiin.

Tiistai 15.5.2018

Päivän tavoitteena oli lisätä nettisivulle uusia asioita sekä viimeistellä tekemiäni front-end-töitä siihen kuntoon, että niitä rohkenee näyttää myös muille. Aamulla oli lyhyt palaveri, jonka jälkeen rupesin töihin.

Eilinen oli uusien ominaisuuksien tekoa, mutta tämä päivä oli niiden viimeistelyä. Osasin viimeistellä koodini yllättävän hyvin ilman dokumentaatioihin katsomista, mikä ei näin jälkeenpäin tarkastellen ole ehkä niin hyvä asia. Dokumentaatioissa olisi varmasti parempia tapoja tehdä asioita. Onneksi ReactJS vaikuttaa olevan aika helposti muokattavissa oleva kieli myös jälkeenpäin.

Tämä päivä oli itse asiassa normaalia päivää hieman tuotteliaampi, vaikken lopulta täysin onnistunutkaan viimeistelemään eilen tekemääni koodia. Viimeistely osoittautui ajateltua työläämmäksi, mutta sain silti paranneltua ominaisuuksia sekä yksinkertaistettua koodiani aika paljon.

Mitä nettisivuihin tulee, niin sain niitäkin hyvän matkaa eteenpäin sekä selvitettyä muutaman ongelmakohdan. Kaiken kaikkiaan olen tyytyväinen tähän päivään.

Keskiviikko 16.5.2018

Tänään tavoitteena oli ratkaista ennalta määrätty ohjelmointipulma sekä tehdä nettisivuja hieman eteenpäin. Päivä oli etäpäivä.

En voi kertoa tarkkoja yksityiskohtia ongelmastani, jota lähdin ratkaisemaan, mutta se oli pohjimmiltaan varsin yksinkertainen. Haasteen toi se, että ongelma ratkaistiin ympäristössä, jossa piti ymmärtää toisen tekemää koodia, mikä tarkoittaa sitä, että kyseinen koodi pitäisi ymmärtää ennen kuin ongelman voi ratkaista. Aluksi tein useitakin noin 20 rivin pituisia suhteellisen monimutkaisia ratkaisuja, jotka toimivat parhaimmillaankin vain osittain. Lopulta sain ongelman ratkaistua vain noin neljällä rivillä koodia. Tässä konkretisoitui se, että mitä vähemmän rivejä oikeasti on, niin sitä parempi yleensä. Päivän oppi ei kuitenkaan ollut välttämättä kyseisen ongelman ratkaisu vaan pikemminkin se, että opin paremmin ymmärtämään toisen tekemää koodia. Nettisivuista ei ole suurempaa kerrottavaa, sillä tein vain perusmuutoksia niihin loppupäivästä.

Jos päivää tarkastelee ulkoapäin tulosten perusteella, niin se olisi varmasti ollut katastrofi, mutta tosiasiallisesti opin tänään aika paljon ja olen päivään tyytyväinen. Osasin ennakoida myös sen, että en tänään varmaankaan paljon muuta ehdi tehdä, ja siksi en asettanutkaan päivän tavoitteita sen korkeammalle kuin ne lopulta olivat.

Torstai 17.5.2018

Tämä päivä oli tavoitteeltaan hieman eilisen oloinen. Tavoitteet ja tehtävät olivat samat. Ongelma vain oli vaihtunut.

Itse konkreettinen ongelma oli tällä kertaa monimutkaisempi, mutta siitä huolimatta perusongelma oli jälleen kerran sama – toisen koodin ymmärtäminen. Ratkaisukin toisti itseasiassa samaa kaavaa: ensiksi tein useita versioita, ja lopuksi yksi toimi. Se ratkaisu, joka toimi, oli yksi yksinkertaisimmista ratkaisuistani kyseiseen ongelmaan. Huomasin eiliseen ratkaisuuni liittyen myös pienen

bugin. Kyseinen bugi ei oikeastaan johtunut suoraan ratkaisustani itsessään vaan muista tekijöistä. Kyseisen bugin korjaus oli kuitenkin suhteellisen helppoa, joten korjasin sen sitten myös samalla.

Nettisivuja tein tänään aika paljon, mutta samoin kuin eilen, tästäkään ei ole paljoa kerrottavaa. Tein tyypillistä määreiden muuttamista ja muuta sellaista, joka kuuluu nettisivujen tekemiseen.

Päivään voi jälleen kerran olla tyytyväinen, sillä päivän tavoitteet saavutettiin ja samalla saatiin bugikin korjattua.

Perjantai 18.5.2018

Tämän päivän teemana oli debuggaus eli virheenetsintä ja -korjaus. Tavoitteena oli viimeistellä nettisivuja ja hieman saada korjattua toista työtäni.

Nettisivuja viimeistellessä ilmeni muutama ongelma, joiden ratkaiseminen vei aikaa, ja kaikkia ongelmia en onnistunutkaan ratkaisemaan. Ongelmat liittyivät pohjimmiltaan sivujen CSS:n rakenteeseen. Jossain on määritelty jotain, joka määrityksen x kanssa aiheuttaa tilan z. Käytännössä näitä ongelmia saadaan debugattua selaimen omalla debuggerilla, jonka saa yleensä päälle F-12-näppäintä painaen. Sivut saatiin lopulta viimeisteltyä suurin piirtein kuten oli tarkoituskin. Käytännössä nettisivut veivät suurimman osan päivästäni ja toiseen työhöni sain vain hieman hahmotelmia siitä, miten asian x voisi tehdä, sekä suunnitelmia, miten jatkaa tästä.

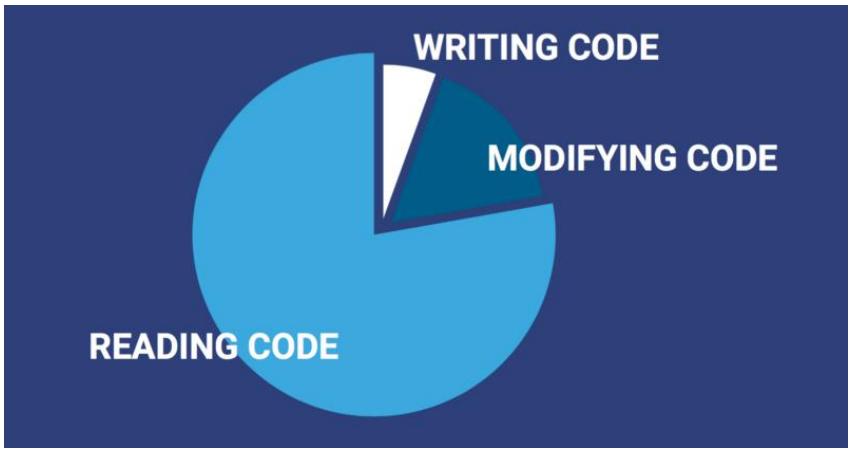
Viikkoanalyysi

Tämäkin viikko oli pitkälti uuden oppimista ja jo olemassa olevan tiedon soveltamista. Erityisesti keskiviikko ja torstai olivat opettavaisia päiviä, sillä yleinen koodinlukutaitoni parani selvästi.

Koodinlukutaito onkin yksi tärkeimmistä taidoista, mikä ohjelmoijalla pitäisi olla. Isoissa sovelluksissa on hyvinkin mahdollista, että työaika menee suurimmaksi osaksi sen ymmärtämiseen, mitä koodissa oikeasti tapahtuu. Gräther kertoo artikkelissaan, kuinka hänellä meni Chromella työskennellessään FPS-mittarin (frames per second) toteuttamiseen kuukausi, vaikka hän aluksi ajatteli sen olevan valmis parissa tunnissa. Syynä oli se, että ohjelmoijan piti ymmärtää hyvin runsas

määrä muuta koodia, jotta kyseistä FPS-mittaria pääsee toteuttamaan. FPS-mittaria ei voinut tehdä millään kopioi ja liitä -menetelmällä keskelle koodia. (Gräther 2017, viitattu 20.5.2018.)

Olen huomannut itse myös, että ohjelmoinnissa kehittyminen ei välttämättä tarkoita koodirivien määrällistä kasvua päivää kohden. Kun ohjelmoinnissa kehittyy, niin ohjelman mahdollisen toiminnan näkee tavallaan mielessä koodin tasolla, ja aika kuluu suunnitteluun ja mahdollisten ongelmakohtien löytämiseen ennen kuin ne ovat edes ongelmia. Luonnollisesti mitä enemmän ohjelmassa on toisen tekemää koodia, sitä vaikeampaa tällaisen mentaalisen mallin luonti on. Emme myöskään ammattikorkeakoulun opintojaksoilla muokanneet erityisen paljon toisten tekemää koodia, tai jos muokkasimme, koodi oli kommentoitu melkein rivi riviltä. Työelämässä kuitenkin harvemmin tehdään näin. Tässä voisi olla myös mahdollisesti ammattikorkeakoulun puolelta parantamisen varaa, kun joskus aikanaan uutta opetussuunnitelmaa laaditaan. Gräther (2017, viitattu 20.5.2018) on laatinut aika mielenkiintoisen kuvion asiaan liittyen (kuvio 4), mutta kuviota tarkastellessa on hyvä muistaa, että ajankäyttö riippuu oikeasti myös paljon henkilöstä ja kokemuksesta sekä projektin ja työn tarkemmasta luonteesta. Kuvio on siis subjektiivinen. Otin kuvion kuitenkin mukaan viikkoanalyysiini, koska koen, että siinä on totuuden siemen ja koska se herättää mielestäni ajatuksia ohjelmoijan ajankäyttöön liittyen. Ohjelmointi ei ole siis ainoastaan koodin kirjoittamista.



KUVIO 4. Grätherin näkemys ohjelmistokehittäjän ajankäytöstä (Gräther 2017, viitattu 20.5.2018)

Maanantai, tiistai ja perjantai sujuivat pitkälti aikaisemmin ammattikorkeakoulussa opituilla taidoilla ja niiden suoralla soveltamisella. Käytännössä tämä tarkoittaa sitä, että pääsääntöisesti kirjoitin koodia enempiä miettimättä. Nyt siis esimerkiksi kuvio 4 ei pätenyt ajankäyttööni. Kriittisesti katsottuna olisin voinut katsoa taitojani ja tekemisiäni hieman lähempää ja selvittää, olisiko tavalle

x löytynyt parempaa vastinetta, jolloin ajankäyttöni olisi muistuttanut kuviota 4. Tällöin olisin varmasti myös oppinut enemmän. Otankin koodin laadun parantamisen ensi viikon teemaksi, vaikka koodia silloin tuleekin vähemmän päivää kohti. Laatu on kuitenkin määrää tärkeämpää pitkällä tähtäimellä. Yleensä on myös parempi, jos ratkaisu on saavutettu muutamalla rivillä verrattuna siihen, että sama ratkaisu on tehty kirjoittaen sivukaupalla koodia. Koodini ei kuitenkaan ole niin sanotusti spagettia, joka on ohjelmoijien käyttämä termi huonolle ja sekavalle koodille, mutta tavoitteena on myös kehittyä ohjelmoijana, eikä se mielestäni onnistu, jos omaan koodiinsa on liian tyytyväinen.

3.3 VIIKKO 21

Maanantai 21.5.2018

Tänään oli tiedossa lyhyt palaveri ja ReactJS:llä koodaamista. Kuten mainitsin viime viikkoraportissani, niin tällä viikolla tavoitteena on pyrkiä parantamaan koodin laatua löytämällä uusia ratkaisumalleja. Siksi otinkin erityisesti tälle päivälle tavoitteeksi parantaa ohjelmistotottumuksiani.

Itse toiminallisuuksien kannalta mentiin tavallaan taaksepäin, sillä suunnittelin aikaisemmin tekemääni koodia hieman uusiksi. Mitään radikaalia ei kuitenkaan tarvinnut tehdä. Tähän oli nyt selvästi paremmat edellytykset kuin aloittaessani työt, eli kehitystä on siis selvästi tapahtunut. Huomasin myös ymmärtäväni ohjelmointikielten teoriasta paljon enemmän. Edellisen viikon viikkoraportin kuvio 4 piti aikataulun kannalta aika lailla paikkaansa. Selvyyden vuoksi mainittakoon, että lasken dokumentaatioiden lukemisen koodin lukemiseksi. Päivän tavoitteet siis saavutettiin. On kuitenkin hyvä muistaa, että koodia ei myöskään kannattane loputtomiin optimoida, sillä silloin se ei valmistu koskaan.

Tiistai 22.5.2018

Tänään oli eräs spesifi tehtävä ReactJS:iä käyttäen. Kyseiseen tehtävään piti lukea aika paljon dokumentaatioita, mutta lopuksi ratkaisu olikin kohtalaisen helppo — tai niin luulin.

Luulin jo tehneeni tehtävän ja siirryin muihin töihin, mutta iltapäivällä selvisikin, että ratkaisussa oli pari isompaa bugia. Koska asia selvisi niin myöhään, ratkaisua bugeihin ei löytynyt päivän päättymiseen mennessä. Tämä on tietyllä tapaa tyypillistä ja tapahtunut myös monta kertaa esimerkiksi ammattikorkeakouluprojektien kanssa. Tämän tyyppiset bugithan voisi estää kunnon testit tekemällä, mutta tässä kyseessä oli kuitenkin rautalankaversio kyseisen annetun tehtävän lopullisesta toteutuksesta eikä mikään lopullinen versio. Testaukset kuitenkin vievät aikaa, ja aika on rajallista. Siitä huolimatta viimeistään lopullisessa versiossa testit on kuitenkin tehtävä.

Sain kuitenkin hyvin tehtyä muuta tehtävää, mikä hieman kompensoi tämän päivän päätehtävän osittaista epäonnistumista. Loppujen lopuksi päivä ei ollut erityisen hyvä eikä huono.

Keskiviikko 23.5.2018

Tänään oli jälleen aika selvät suunnitelmat tehtävistä mitä tehdä, joiden työstäminen oli myös tavoitteeni. Rupesin tekemään tehtäviä, mutta iltapäivällä huomasin törmänneeni ongelmaan, jonka yksityiskohtia en voi avata enempää. Käytännössä esteenä olivat kuitenkin oma osaaminen sekä muut tekijät koodissa. Itse asiassa minunkin täytyy hieman selvitellä yksityiskohtia ja sitä, miten tästä edetään.

Minulla on kuitenkin kohtuullisen vapaat kädet työni suhteen, joten rupesin tekemään toista projektia, ja sen suhteen onnistuinkin ihan hyvin. Sain tehtyä järkeviä ominaisuuksia, joita mahdollisesti hion huomenna lisää. Alkuperäisiä tavoitteita ei saavutettu, mutta olen päivään silti tyytyväinen, sillä sain jotain järkevää kuitenkin siitä huolimatta tehtyä.

Torstai 24.5.2018

Tänään tavoitteena oli viimeistellä eilisen alkuperäisestä suunnitelmasta poikenneet ominaisuudet. Kuten oli odotettua, alku alkoi hieman nihkeästi, mutta loppupäivästä löysin kohtuullisen hyvät ratkaisut muutamaan bugiin, jotka ominaisuuksissani oli. En ehtinyt kuitenkaan täysin toteuttaa ratkaisuani, mutta en usko, että sen kanssa tulee suurempia ongelmia.

Muita ongelmia tässä päivässä kuitenkin riitti. Versiohallinnan kanssa tuli hieman säätöä kesken muutostöiden, mikä johtui vääristä komennoista. Onneksi pahimmat ongelmat koskivat irrelevant-

tia dataa, joka oli joka tapauksessa tallessa. Se opetti kuitenkin sen tärkeydestä, että varmuuskopiot on aina säilytettävä myös lokaalisti eli esimerkiksi omalla tietokoneella. Teen kuitenkin näin onneksi jo nyt. Yleensä poistan varmuuskopiot vasta, kun ne lakkaavat olevasta relevantteja. Pelkkä versiohallinta ei yksinkertaisesti riitä varmuuskopioksi, jos haluaa olla varma siitä, että tiedot pysyvät tallessa. Kannattaa harkita, riittääkö edes kaksi paikkaakaan, jos tieto on oikeasti tärkeätä. Monesti varmuuskopiot ovat paikassa, jota ei käytetä paljon, jolloin siitä, onko tieto kadonnut, ei voi olla varma. Tällainen vaarallinen paikka voi olla esimerkiksi ulkoinen kovalevy tai muistitikku, jota käytetään vain vahingon sattuessa. Muistitikkuja ei välttämättä voi kyllä edes laskea varmuuskopioksi, sillä ne hajoavat ainakin omien kokemuksieni mukaan todella helposti.

Tähän päivään ei kuitenkaan voi olla erityisen tyytyväinen, vaikka opinkin tänään paljon. Loppujen lopuksi tavoitteet, jotka eivät olleet edes kovin korkealla, jäivät saavuttamatta, ja lisäksi ilmeni ylimääräisiä ongelmia, jotka eivät ole koskaan kovin mukavia.

Perjantai 25.5.2018

Tänään oli palaveripäivä, ja päivän tavoitteet oikeastaan liittyivätkin siihen. Otin kannettavalle tuotokseni, jotta pystyn esittelemään niitä ja viimeistelin niitä myös hieman. Tein aamulla myös muita pienempiä töitä liittyen koodin siistimiseen. Mitään suurempaa en tänään kuitenkaan ohjelmoinut.

Palaveri oli oikeastaan aika tyypillinen tilannekatsaus, mutta en mene sen enempää yksityiskohtiin. Sain kuitenkin hyviä vinkkejä ja kuulin esimerkiksi ReactJS:ään liittyen uusista ominaisuuksista, joita aion koettaa heti ensi viikolla käytännössä.

Päivä meni siis aika odotetusti, eikä mitään yllättävää tapahtunut suuntaan tai toiseen. Mukava rauhallinen lopetus viikolle.

Viikkoanalyysi

Viikko ei todellakaan ollut mikään täydellinen muttei kyllä katastrofikaan, sillä työt etenivät kuitenkin selvästi. Viikko oli myös erittäin opettavainen. ReactJS:n syntaksi alkaa jo tuntumaan oikeasti luonnollisemmalta, ja sillä osaa tehdä nyt pitkälti samoja asioita kuin muillakin kielillä. Lisäksi git-versionhallinta tuli tutummaksi. Luulin tosin jo osaavani gitin aika hyvin, mutta tämä viikko todisti luuloni vääräksi. Aina löytyy uusia komentoja ja ominaisuuksia. Sama oli oikeastaan CSS:n

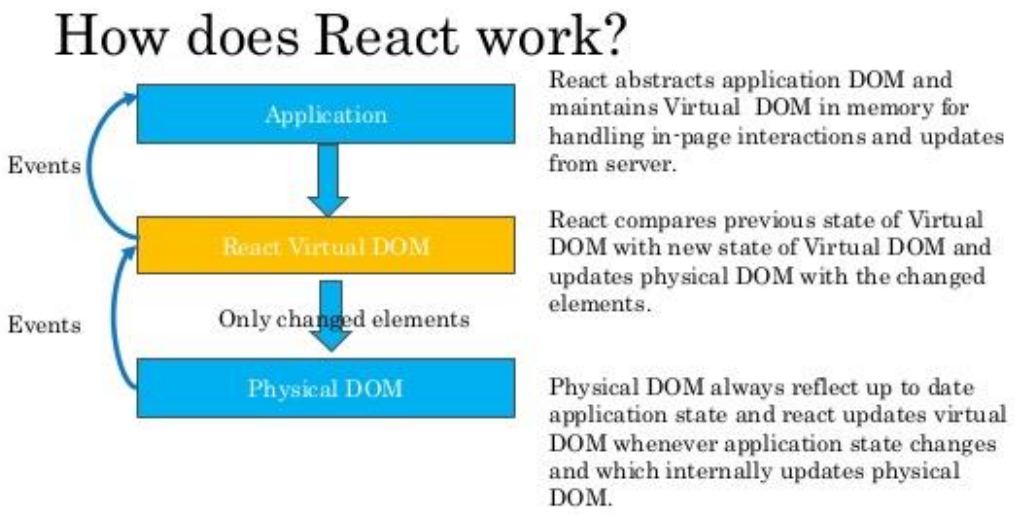
kanssa aikoinaan. Luulin joskus osaavani sitä erittäinkin hyvin, mutta kohta tämäkin osoittautui vääräksi, koska myös sillä on monimutkaisia sovellutuksia. CSS:n kanssa ei kuitenkaan toistaiseksi ole ollut isompia ongelmia töissä, vaikka en toisaalta sillä olekaan toistaiseksi tehnyt mitään erityisen vaikeaa.

Viikkokirjoituksissani ovat jääneet vähemmälle huomiolle erilaiset palaverit, joita on ollut kuluneen kolmen viikon aikana. Ne vievät kuitenkin ison osan työajasta ja ovat olennainen osa työnkuvaa. Palavereissa omat tekemiset selkenevät muille ja itselle selkenee se, mitä tehdä jatkossa. Palavereiden merkitys korostuu myös siksi, että teen suurimman osan työstäni etänä, jolloin ihmisiä ei välttämättä näe ja on vaikeampi saada kokonaiskuvaa tilanteesta. Lisäksi palavereissa oppii joskus myös jotain uutta. Esimerkiksi perjantaina sain koodiin pari käyttökelpoista vinkkiä, joita en olisi varmasti googlaamalla löytänyt. Vinkkien yksityiskohtia en voi sanoa, sillä ne paljastaisivat liikaa siitä, mitä olen tarkalleen tekemässä. Olen kuitenkin huomannut, että työelämässä palavereista on oikeasti hyötyä, vaikka opinnoissa ne tuntuivat lähinnä ajanhukalta. Syy varmaan on siinä, että ammattikorkeakoulussa muut projektin jäsenet olivat kaikki oikeasti jo tuttuja osittain myös vapaa-ajalta, jolloin tieto vaihtuu lähes automaattisesti.

Todellinen viikon tavoite oli kuitenkin oppia kirjoittamaan parempaa ReactJS:n koodia, ja se tavoite kyllä saavutettiin. Kuten aikaisemmin mainitsin, niin syntaksi alkaa olemaan jo tutumpi, jolloin resursseja voi käyttää itse ongelmien ratkomiseen. Muutaman kerran on ollut tilanne, että tietäisi miten ongelma x ratkeaisi kielellä y, mutta ReactJS:ää käyttäen ei ole aavistustakaan siitä, miten kyseinen ratkaisu tehdään sillä.

Suurin ongelma on ollut yrittää opetella, miten ReactJS:ssä muutetaan DOMia (Document Object Model). Se tehdään virtuaalisesti, mikä tarkoittaa, ettei DOMia muokata suoraan, kuten esimerkiksi jQueryssä tehdään. Käytännön koodissa se näkyy erilaisena syntaksina, siten että se viittaa suoraan DOM:iin. Vaikka ReactJS olisikin käytössä, niin DOMiin voi silti viitata periaatteessa suoraan, mutta niin ei todellakaan tulisi tehdä konfliktien välttämiseksi. Tein ihan alussa välillä näin, mutta olen jo muuttanut sen aikaiset koodit ReactJS:n mukaisiksi. Otan kuitenkin seuraavan viikon tavoitteeksi virtuaalisen DOMin syvällisemmän ymmärtämisen, sillä vaikka osaankin jo soveltaa sitä käytäntöön, niin pelkkä soveltaminen ei riitä vaan välillä vaaditaan syvällisempää asiantuntijuutta. Kuviossa 5 on kuvattu lyhyesti ReactJS:n toimintaa tarkemmin.

Virtuaalisella DOMilla saavutetaan nopeushyötyjä varsinkin, jos sovellus on suuri ja raskas. ReactJS on erityisen hyvä SPA:n teossa (single page application). ReactJS:llä voi kuitenkin tehdä muutakin kuin SPA:lla. On hyvä tietää myös se, että jos ReactJS:ää verrataan MVC-malliin (model, view, controller), niin se keskittyy ainoastaan V-osioon eli näkymiin. (Palakollu 2016, viitattu 28.5.2018.)



KUVIO 5. ReactJS:n toiminta (Palakollu 2016, viitattu 27.5.2018)

3.4 VIIKKO 22

Maanantai 28.5.2018

Tänään tavoitteena oli tehdä uusia ominaisuuksia ja parannella jo olemassa olevia ominaisuuksia ohjelmistoon. Pääohjelmointikieleni on edelleen ReactJS. Kuten viime viikkoraportissa sanoin, niin tarkoituksena on oppia ymmärtämään työtäni paremmin myös teorian tasolla, sillä se parantaa myös työn jälkeä. Mielessä on luonnollisesti pidettävä myös se, että siitä olisi oltava oikeaa hyötyä myös työnantajalle, sillä muuten se kuuluu enemmän vapaa-aikaan. Tänään kuitenkin käytännön ohjelmointityö sujui niin hyvin, ettei suuremmille lukemisille ollut aikaa, jos aivan perus dokumentaatioiden lukemista ei lasketa. On taottava, kun rauta on kuumaa, sillä tällainen flow ei ole ihan jokapäiväistä. Saavutin päivän tavoitteet kirkkaasti ja hieman ylitinkin ne. Tekemätöntä

työtä tietysti jäi, sillä harvemmin päivässä kuitenkaan mitään ihmeitä saa aikaan. Tällaisina päivinä silti huomaa, että jotain on myös oppinut. Pitää toivoa, että flow pysyisi yllä koko viikon, sillä tiedossa on myös aika haastavia tehtäviä.

Tiistai 29.5.2018

Tänään päivä oli tavoitteiden osalta identtinen eilisen kanssa, eli tavoitteena oli parantaa jo olemassa olevia ominaisuuksia, sekä luoda uusia ominaisuuksia ja integroida niitä jo olemassa oleviin ominaisuuksiin. Tänäänkin oli aika tuottava päivä, enkä erityisemmin nähnyt järkeä ruveta lukemaan teoriaa. Ehkä jätän teoriat jollekin toiselle päivälle, jolloin voin käyttää teoriaa apuna jonkun isomman ongelman ratkaisemiseen. Tänäänkin tosin ratkesi eräs suurempi viime viikolla useita tunteja vaivannut ongelma. Ratkaisu oli pohjimmiltaan yhden rivin pituinen.

En sanoisi kuitenkaan, että päivä oli ihan eilisen tyylinen, mutta se oli silti erittäin hyvä ja ehdottomasti normaalia parempi. Tavoitteet saavutettiin kirkkaasti, vaikka olisin halunnut päivän lopuksi vielä viimeistellä pari ominaisuutta. Väsyneenä koodia ei kuitenkaan mielestäni kannata kirjoittaa, sillä muuten sen saa kirjoittaa kaksi kertaa: väsyneenä ja virkeänä.

Keskiviikko 30.5.2018

Tänään tavoitteena oli viimeisteillä maanantaina ja tiistaina tehtyjä ominaisuuksia, sekä tehdä hieman muuta ennalta sovittua työtä. Ennalta sovittuun työhön sovelsin viime perjantaina saamiani ohjeita. Siinä tuli pieniä ongelmia, joiden selvittelemiseen kului pari tuntia, mutta kun sain koodin toimimaan, kaikki meni kuten suunniteltu. Uskon, että pystyn soveltamaan tässä oppimaani myös jatkossa.

Viimeistely oli hieman isomman prioriteetin työ, ja siinä tavoitteet myös saavutettiin. Voin aika ylpeänä esitellä nyt alkuviikon tuotokseni. Toiminnallisuuksissa on hiomisen varaa, toisaalta luulen ainakin tietäväni, että miten kaikki toiminallisuuksiin liittyvä koodi tehdään. Todellisuudessa luulo ei ole tiedon väärti ja ongelmia on varmasti tiedossa, mutta niiden ratkominen on monesti jopa ihan mukavaa puuhaa.

Kaiken kaikkiaan päivä oli aika tasaisen oloinen, tavoitteet saavutettiin, mutta juuri mitään ylimääräistä en saanut aikaiseksi.

Torstai 31.5.2018

Tänään oli tiedossa hieman haastavampaa työtä, josta myös oli sovittu ennalta jo aikaisemmin. Tässä haasteen toi toisen koodin ymmärtäminen. Minulla meni yhtään liioittelematta useita tunteja siihen, että hahmottelin koodin rakennetta paperille. Paperi on hyvä apuväline, jos täytyy ymmärtää usean eri asian suhdetta toisiinsa. Tähän tarkoitukseen löytyisi kyllä ohjelmiakin, mutta olen kai hieman vanhanaikainen, kun suosin kynää ja paperia. Päivän tavoitteena oli tehdä sekä viimeistellä kaksi eri ominaisuutta.

Toisen ominaisuuksista onnistuin tekemään, sekä viimeistelemään kokonaan. Toinen ominaisuus kuitenkin jäi puolitiehen. Lisäksi koetin muokata jo aikaisemmalla viikolla tehtyä ominaisuutta tarkemmin spekseihin sopivammaksi. Siinä pääsin lähelle monellakin eri lähestymistavalla, mutta ratkaisu jäi tekemättä. Kyseessä voi hyvinkin olla logiikkavirhe, joka yleensä ratkeaa seuraavana päivänä.

Päivä oli kuitenkin mielestäni ihan hyvä, sillä epäilinkin, että tämän päivän tavoitteet jäävät saavuttamatta osittain, ja lisäksi pelkäsin, etten saisi mitään aikaiseksi. Sain kuitenkin jotain konkreettista tehtyä sekä opittua uutta asiaa, johon voi olla aika tyytyväinen lopulta kuitenkin.

Perjantai 1.6.2018

Tänään oli tiedossa palaveri puolilta päivin ja päivän tavoitteet liittyivätkin siihen, että saisin sieltä taas hieman uutta tietoa ja muistaisin kertoa itse kaikki olennaiset asiat. Aamulla valmistauduin palaveriin viimeistelemällä viime viikon töitäni ja siistimällä kuolleet koodit ja muut ylimääräiset tekijät. Tein aamulla myös hieman sitä mihin torstaina jäin, mutta en päässyt kovinkaan pitkälle, sillä päivä tuntui kuluvan niin nopeasti.

Palaverissa kuitenkin selvisi, missä olin mennyt vikaan torstaina. Kyseessä oli ilmeisesti jälleen aika yksinkertainen vika uskoakseni, mutta en toteuttanut korjausta kuitenkaan loppuun, sillä keksin yhtäkkiä mistä eräs puolessa välissä viikkoa ilmestynyt suhteellisen iso ja ärsyttävä bugi johtui. Kyseinen bugi ilmestyy vain joskus ja tietyissä tilanteissa. Selvitin mitä todennäköisimmän syyn ja tein siihen korjauksen, mutta kyseinen korjaus on vielä hieman vajavainen, olen varmasti kuitenkin oikeilla jäljillä ja käyn korjaamassa, jos jokin parempi ratkaisu tulee mieleen. Päivä oli ihan hyvä, ja voisin uskoa, että tänään opittuja asioita voidaan käyttää ensi viikolla hyväksi.

Viikkoanalyysi

Viikko oli ehkä tehokkain viikko tähän mennessä. Opin tälläkin viikolla paljon uutta asiaa, vaikka varsinkin alkuviikko meni enemmänkin osaamista soveltaessa kuin uutta oppiessa.

Mutta jos tätä viikkoa vertaa edellisiin, vaikkakin työtehtävät sinällään ovat olleet aika samantyyppisiä, niin voisi sanoa, että kaksi ensimmäistä viikkoa koostui lähinnä uuden oppimisesta, kolmas ehkä yrityksen ja erehdyksen kautta oppimisesta ja nyt tämä viikko oli lähes aitoa tiedon soveltamista. Tämä on ilmaistu siis hieman kärjistäen, sillä olen kyllä soveltanut esimerkiksi ammattikorkeakoulussa oppimaani joka päivä. Sitä ei vain aina tule edes ajatelleeksi, että esimerkiksi ohjelmointikielten perusrakenne on opittu ammattikorkeakoulussa. Esimerkkinä asiat, kuten for-silmukka tai if-lauseet, on opittu ammattikorkeakoulussa, ja ne tulevat ihan luonnostaan, mutta kyllä nekin on aikanaan pitänyt opiskella ja, jos niitä ei osaisi, niin en usko, että ohjelmointi olisi edes mahdollista.

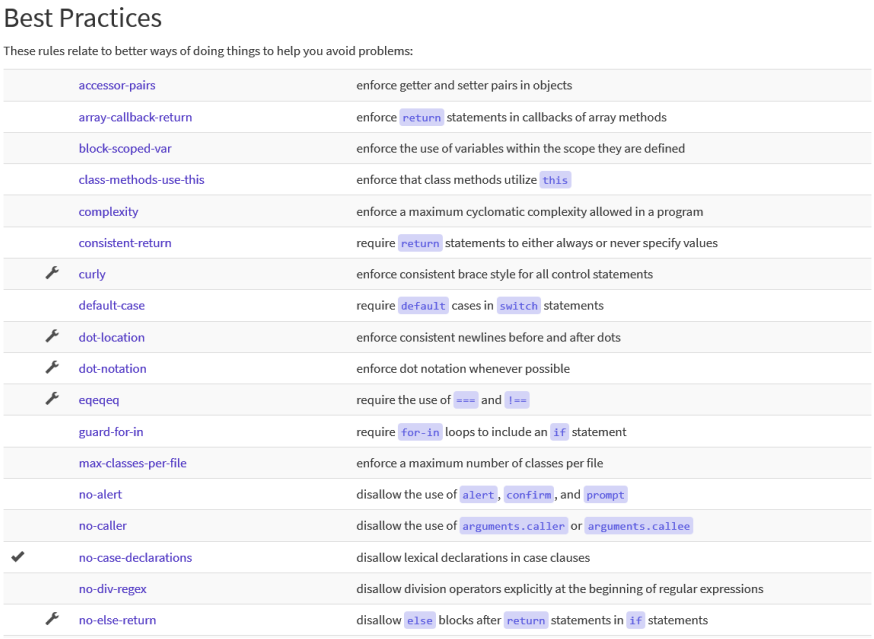
Olen aikaisemmilla viikoilla puhunut koodin laadusta ja siitä, että siinä kehittyminen on yksi ohjelmoijan suurimmista ja tärkeistä haasteista. Koodin on oltava selkeää, luettavaa ja mahdollisimman yksinkertaista aina. Näitä asioita selvittäessä ja opiskellessa olen törmännyt useasti käsitteeseen ”best practices” (parhaat käytänteet). On olemassa yleisiä käytänteitä, jotka pätevät yleisesti, ja käytänteitä, jotka ovat ohjelmointikielikohtaisia.

Yleisiä parhaita käytänteitä voivat olla esimerkiksi yhtenäiset sisennykset, liian ilmiselvien kommenttien pois jättäminen, koodin toiston minimointi, liiallinen koodin koteloinnin välttäminen (nesting) tai horisontaalisesti liian pitkien koodin tekeminen. (Guzel 2011, viitattu 3.6.2018.) Itse tunnistan kehittämiskohteeksi liiallisen koteloinnin ja horisontaalisesti pitkät rivit. Tämä johtuu siitä, että kun koodin saa toimimaan potentiaalisesti pitkän taistelun jälkeen, siinä on aika korkea kynnys lähteä muokkaamaan sitä.

Parhaisiin käytänteisiin on olemassa myös työkaluja, joiden avulla voi varmistaa, että toimii uusimpien parhaiden käytänteiden mukaan. Yksi tällainen työkalu on ESLint, joka on työkalu, joka on erikoistunut JavaScriptin koodin laadun parantamiseen esimerkiksi juuri mainittujen parhaiden käytänteiden avulla. En itse käytä ESLintiä, mutta se näyttää kyllä aika mielenkiintoiselta työkalulta, ja mahdollisesti alan käyttämään sitä jossain vaiheessa ainakin omissa henkilökohtaisissa

projekteissani. Vaikka sitä ei nyt sitten lopulta käyttäisikään, niin vähintäänkin tällaisten työkalujen dokumentaatiosta saa vinkkejä oman henkilökohtaisen ohjelmointitavan parantamiseen.

ESLintin dokumentaatioista pystyy lukemaan asiat, joita kyseinen työkalu pitää parhaina käytänteinä. Kuviossa 6 on osa ESLintin parhaista pitämistä käytänteistä. Niitä voi myös konfiguroida pois, eikä ESLint pakota käyttämään kyseisiä käytänteitä, ellei niin halua. (ESLint 2018, viitattu 3.6.2018.)



Best Practices

These rules relate to better ways of doing things to help you avoid problems:

	accessor-pairs	enforce getter and setter pairs in objects
	array-callback-return	enforce `return` statements in callbacks of array methods
	block-scoped-var	enforce the use of variables within the scope they are defined
	class-methods-use-this	enforce that class methods utilize `this`
	complexity	enforce a maximum cyclomatic complexity allowed in a program
	consistent-return	require `return` statements to either always or never specify values
🔧	curly	enforce consistent brace style for all control statements
	default-case	require `default` cases in `switch` statements
🔧	dot-location	enforce consistent newlines before and after dots
🔧	dot-notation	enforce dot notation whenever possible
🔧	eqeqeq	require the use of `===` and `!==`
	guard-for-in	require `for-in` loops to include an `if` statement
	max-classes-per-file	enforce a maximum number of classes per file
	no-alert	disallow the use of `alert`, `confirm`, and `prompt`
	no-caller	disallow the use of `arguments.caller` or `arguments.callee`
✔	no-case-declarations	disallow lexical declarations in case clauses
	no-div-regex	disallow division operators explicitly at the beginning of regular expressions
🔧	no-else-return	disallow `else` blocks after `return` statements in `if` statements

KUVIO 6. ESLint Best Practices

3.5 VIIKKO 23

Maanantai 4.6.2018

Tänään tavoitteena oli aloittaa muutama isompi ominaisuus, jotka toivottavasti saataisiin hiottua loppuviikkoon mennessä jonkinlaiseen muotoon. Aamulla oli jälleen yksi bugi, jonka halusin myös korjata ennen tämän päivän urakkaa. Katselin pari tuntia kyseistä bugia ja sain hieman rajattua sitä, jolloin ajattelin, että on hyvä yrittää tehdä muutakin. Rupesin siis tekemään pohjaa isommille ominaisuuksille, ja suurempia ongelmia sen tekemisen kanssa ei oikeastaan ilmaantunut.

Illalla luin netistä ohjelmointiaiheisia foorumeita ilman suurempia tavoitteita ja löysin sattumalta ratkaisun tähän aamupäivän bugiin. On aika jännää sinällään, että ratkaisun löytää sattumalta,

mutta kun etsii, niin mitään sovellettavaa ei löydy. Olen yleisestikin huomannut, että ammattikorkeakouluprojekteihin tiedon etsiminen oli paljon helpompaa töihin verrattuna. Toki ammattikorkeakoulussa tilanne oli se, että kun saman ongelman parissa teki töitä 20 muuta henkilöä, niin kaverilta sai kyllä aina nopeasti apua. Se ei ehkä aina välttämättä kuitenkaan ole hyvä asia, sillä parhaiten juuri oppii, kun asian pohtii kunnolla.

Tiistai 5.6.2018

Tänään tavoitteena oli siistiä koodia, mikä käytännössä tarkoitti koodin kommentointia, yksinkertaistamista, kuolleen koodin poistoa ja sen varmistamista, että asioita ei ole kirjoitettu moneen kertaan. Tavoite saavutettiin yllättävän nopeasti, ja jäljelle jäi vielä puoli päivää.

Loppupäivä menikin sitten eilen aloitettujen ominaisuuksien kehityksessä. Mitään maata mullistavaa ei kuitenkaan saatu tehtyä, vaan ominaisuudet etenivät jonkin verran ja oikeastaan työ olikin enemmän pohjaa tulevalle viikolle. Päivän tavoite kuitenkin saavutettiin kevyesti, joten siksi päivään voi olla tyytyväinen.

Keskiviikko 6.6.2018

Tänään aamulla tiedossa oli speksausta eli eräänlaista ohjelmiston määrittelyä. Päivän tavoitteet liittyivät tähän, koska väärillä spekseillä asiat täytyy tehdä moneen kertaa. Tästä en kuitenkaan voi puhua näin mainintaa enempää. Käytännössä päivä meni pitkälti kyseisen työn parissa.

Illemmalla aloin luomaan nopeaa prototyyppiä aamulla sovituista asioista, jolloin kokonaiskuva paranee ja voidaan sopia, jatketaanko tällä linjalla, vai muutetaanko linjaa. Päivän tavoitteet saavutettiin, vaikka prototyyppi jäikin hieman puolitiehen, mutta se oli oikeastaan odotettavissa, sillä aika oli tänään tiukalla.

Torstai 7.6.2018

Tämä päivä oli oikeastaan kuin kopio eilisestä eli speksausta ja ohjelmointia, painottuen kuitenkin ohjelmointiin. Esittelin myös erilaisia ideoita ja eilen tehtyä prototyyppiä, joka siis oikeastaan oli tehty juuri esittelyä varten. Päätavoitteena oli tänään kuitenkin oikeastaan viedä eilisen aloitettuja töitä eteenpäin. Sain palautetta ja rupesin sen perusteella suunnittelemaan ja tekemään tarvittavaa koodia ominaisuuksille.

Pienimmät ja yksinkertaisimmat kokonaisuudet saatiin tehtyä jo tänään, mutta isompia kokonaisuuksia jäi luonnollisesti tekemättä. Lisäksi mukana on pari ongelmaa, jotka pitää selvittää, jotta ominaisuuksien teossa pääsee eteenpäin. Ongelmat eivät kuitenkaan ole mitään tavallisuudesta poikkeavia, ja veikkaisinkin, että ne ratkeavat pian. Tyypillisesti tämän tyyppiset ongelmat ratkeavat yksinkertaisesti sillä, että tarkastelee samaa ongelmaa eri päivänä.

Perjantai 8.6.2018

Asetin päivän tavoitteeksi erään isomman ominaisuuden tekemisen. Lisäksi tänään oli tiedossa palaveri. Ominaisuus jäi hieman kesken. Sain sen toimimaan selaimen konsolissa halutulla tavalla mutta en kuitenkaan itse ohjelmassa. Päivällä ollut palaveri oli tänään aika lyhyt tilannekatsaus. Palaverin jälkeen jatkoin aamulla aloitettua ominaisuuden tekemistä mutta en juurikaan edennyt. Päivän tavoitetta ei saavutettu, mutta koska edistystä tapahtui, niin en ole päivään kuitenkaan erityisen pettynyt. Uskon, että kun seuraavan kerran jatkan tästä, niin saan tälle päivälle olleet tavoitteet saavutettua siinä samalla.

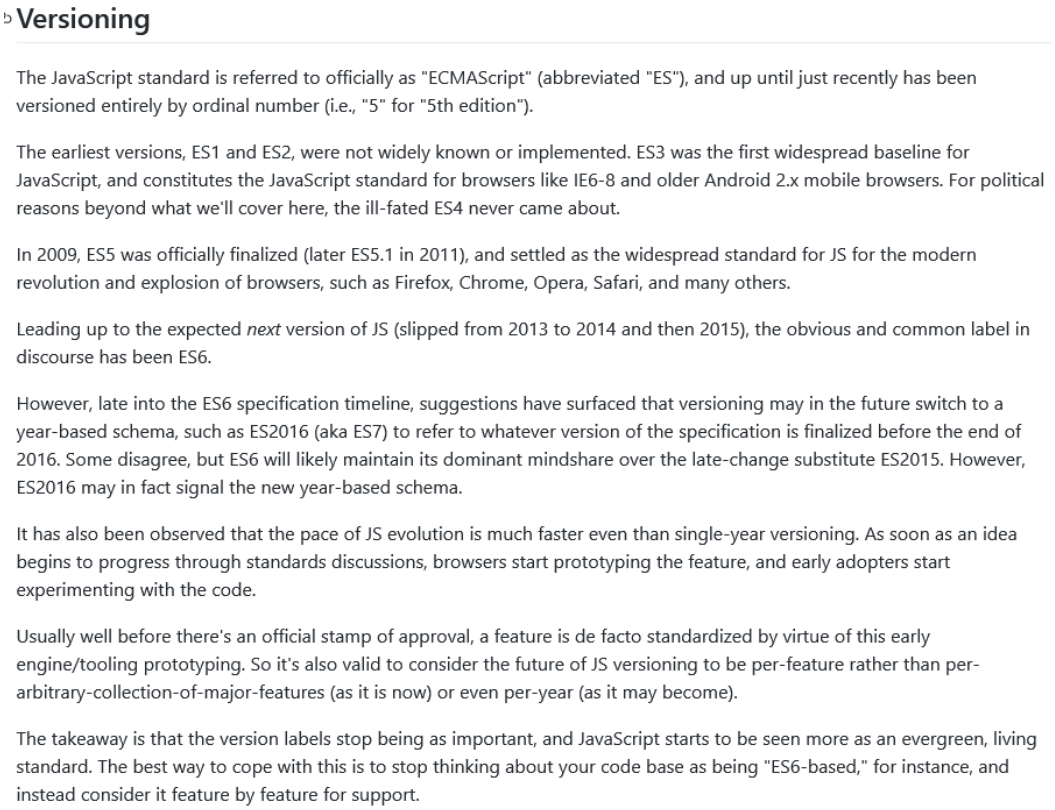
Viikkoanalyysi

Tämä viikko erosi muista viikoista hieman, sillä viikolla oli paljon erilaisia palavereita ja muuta sellaista. Osaaminen on kuitenkin selvästi kehittynyt, koska sain aikaiseksi siitä huolimatta lähes yhtä paljon kuin sellaisellakin viikolla, jolla palavereita ei olisi ollut näin paljon. Tämä ei välttämättä tule kuitenkaan tämän viikon päiväanalyyseistä täysin esille, sillä tavoitteet olivat myös korkeammalla ja myös siksi osa tavoitteista jäi puolitiehen. Jos vertaa ensimmäisiin viikkoihin, niin

uskaltaisin väittää, että teen nyt noin kaksi kertaa nopeammin koodia, joka on kaiken lisäksi parempaa koodia. Nykyisessäkin toki on vielä huimasti parannettavaa.

Olen tosiaan yhä työskennellyt pääsääntöisesti ReactJS:n kanssa. ReactJS on kuitenkin JavaScriptiä, joten ReactJS:ssä kehittyminen tarkoittaa myös JavaScriptissä kehittymistä. JavaScriptille on lisäksi olemassa kymmeniä muita kirjastoja ja frameworkeja, joten JavaScript itsessään on oikeasti se tärkein kieli, missä pitäisi kehittyä.

Löysin hyvän kirjan, jossa JavaScriptiä käydään aika kattavasti läpi. Kirjan nimi on You-Dont-Know-JS ja se löytyy osoitteesta https://github.com/getify/You-Dont-Know-JS/. Nimi viittaa siihen, että JavaScriptissä on niin paljon asiaa, että todella harva osaa sitä täydellisesti. Puhuinkin tästä asiasta viikon 21 viikkoanalyysissä, jossa mainitsin, että luulin osaavani gitin ja myös CSS:n aikanaan, mutta luulo ei ollutkaan tiedon väärti. Kirja on vielä kesken, mutta olen lukenut async & performance sekä es6 & beyond -osioita ja kokenut saaneeni niistä jo paljon irti. Kirja on kirjoitettu vuonna 2015, joten siinä ei ole kaikkia uusimpia asioita, mutta mielestäni se on silti varsin ajan tasalla. Sitä on myös committien perusteella päivitetty hieman myös viime aikoina. Erityisen mielenkiintoiseksi koin ES:n (ECMAScript) historian. Kuviossa 6 on lyhyt katkelma siitä. ES on siis JavaScriptin standardi. Ensimmäisillä viikoilla tiesin vain, että jokin tällainen on olemassa, mutta en oikeastaan sitä, mitä se tarkoittaa käytännössä. Kirjassa oli toki myös paljon muuta mielenkiintoista, mutta jos asia oikeasti kiinnostaa, niin suosittelen ehdottomasti tutustumaan kyseiseen kirjaan. (Getify 2015, viitattu 10.6.2018.)



Versioning

The JavaScript standard is referred to officially as "ECMAScript" (abbreviated "ES"), and up until just recently has been versioned entirely by ordinal number (i.e., "5" for "5th edition").

The earliest versions, ES1 and ES2, were not widely known or implemented. ES3 was the first widespread baseline for JavaScript, and constitutes the JavaScript standard for browsers like IE6-8 and older Android 2.x mobile browsers. For political reasons beyond what we'll cover here, the ill-fated ES4 never came about.

In 2009, ES5 was officially finalized (later ES5.1 in 2011), and settled as the widespread standard for JS for the modern revolution and explosion of browsers, such as Firefox, Chrome, Opera, Safari, and many others.

Leading up to the expected *next* version of JS (slipped from 2013 to 2014 and then 2015), the obvious and common label in discourse has been ES6.

However, late into the ES6 specification timeline, suggestions have surfaced that versioning may in the future switch to a year-based schema, such as ES2016 (aka ES7) to refer to whatever version of the specification is finalized before the end of 2016. Some disagree, but ES6 will likely maintain its dominant mindshare over the late-change substitute ES2015. However, ES2016 may in fact signal the new year-based schema.

It has also been observed that the pace of JS evolution is much faster even than single-year versioning. As soon as an idea begins to progress through standards discussions, browsers start prototyping the feature, and early adopters start experimenting with the code.

Usually well before there's an official stamp of approval, a feature is de facto standardized by virtue of this early engine/tooling prototyping. So it's also valid to consider the future of JS versioning to be per-feature rather than per-arbitrary-collection-of-major-features (as it is now) or even per-year (as it may become).

The takeaway is that the version labels stop being as important, and JavaScript starts to be seen more as an evergreen, living standard. The best way to cope with this is to stop thinking about your code base as being "ES6-based," for instance, and instead consider it feature by feature for support.

KUVIO 6. ECMAScript versiointi (Getify, viitattu 10.6.2018)

Seuraavalla viikolla otankin toimintamalliksi yrittää ymmärtää paremmin JavaScriptiä ja pyrin sitä kautta ymmärtämään esimerkiksi ReactJS:ää paremmin. Seuraavan viikon työtehtävät vaikuttavat toki myös hieman siihen, että mitä tosiasiallisesti teen. Tämä viikko oli varsin onnistunut, vaikka itse ohjelmointia olikin normaalia viikkoa vähemmän. Ohjelmoijan työ ei kuitenkaan ole täysin pelkkää ohjelmointia.

3.6 VIIKKO 24

Maanantai 12.6.2018

Tänään oli tarkoitus viedä loppuun pari isompaa kokonaisuutta. Päivä oli kuitenkin rehellisesti sanoen aika tahmainen. Isommissa kokonaisuuksissa tuntui tulevan seinä vastaan, joten tein pari pienempää tehtävää ja yritin etsiä apua dokumentaatioista, YouTube -videoista sekä erilaisilta foorumeilta. Onnistuin yhden isomman kokonaisuuden viemään loppuun, mutta se oli oletettuakin, sillä tiesin jo etukäteen, mitä tehdä. Tietoa etsiessäni opin kuitenkin paljon ja uskon, että pystyn huomenna soveltamaan tänään opittua tietoa paremmin. Päivän tavoitteita ei siis saavutettu,

mutta tehtävissä edettiin, vaikkakaan ei haluttua tahtia. Kokemukseni mukaan ohjelmoijan arki on monesti kuitenkin juuri tällaista vuoristorataa.

Tiistai 13.6.2018

Tänään tavoitteena oli tehdä asioita, joita eilen jäi puolitiehen ja mahdollisesti jotain pienempää ennalta määrittelemätöntä vielä siihen päälle. Koska ongelmiini ei löytynyt netistä oikein ratkaisuja, niin koetin erilaisia koodiviritelmiä. Yksi viritelmistä näytti lupaavalta, joten jatkoin sen jatkokehitystä. Tässä ongelmanratkaisukeinona oli siis eräänlainen summassa ammunta, jolla monesti saadaan ratkaisu, vaikkakaan ei parasta. Se, mistä tiedän, alkaako jokin ratkaisu toimimaan vai ei, tulee varmaankin kokemuksesta, sillä käytännössä se on vain jonkinlainen tunne, jota on vaikea selittää. Ratkaisukeinoni kuitenkin toimi, ja sain isoimman ongelman tällä ratkaistua. Sen ratkaisun myötä myös muut ongelmat ratkesivat kuin itsestään, ja uskoisin, että eilisen lukemiset kontribuoivat tähän merkittävästi.

Saavutin tänään eilisten tavoitteiden lisäksi myös paljon muuta asiaa, jonka luulin olevan paljon hankalampaa, ja siksi olinkin varannut niille enemmän aikaa. Ainoa, mihin en tässä päivässä ollut tyytyväinen, oli se, että en ehtinyt siivota koodia kaikista konsolikirjauksista ja testikoodeista. Tämä tapahtui siksi, koska olin päivän loputtua niin väsynyt, että en yksinkertaisesti jaksanut. Tiedossa oli myös opinnäytetyöni ohjausseminaari päivän jälkeen. Vasta toimivaksi tehtyä koodia ei koskaan muutenkaan kannata sorkkia liikaa, jos on oikeasti väsynyt. Päivään voi kuitenkin olla varsin tyytyväinen.

Keskiviikko 14.6.2018

Tämän päivän tavoite on itse asiassa yhteinen tavoite myös torstaille ja perjantaille. Tavoitteena on tehdä kolme isompaa kokonaisuutta erääseen ohjelmaan. Arvelen, että onnistun toteuttamaan kaksi ja yksi jää puolitiehen, mutta olisin myös siihen varsin tyytyväinen. Tänään rupesin tekemään mielestäni toisiksi vaikeinta kokonaisuutta näistä vaihtoehdoista. Syynä on se, että minulla oli omia teorioita valmiina kyseisen kokonaisuuden toteuttamiseen.
Käytännössä teoriat toimivat osittain. Sain ominaisuudesta ydintoiminnot toimimaan, mutta erääseen kohtaan jäi yksi isompi bugi, joka pitäisi ratkaista. Kyseisen bugin voi kuitenkin helposti

kommentoida, jolloin ominaisuus toimii hieman vajavaisempana versiona. Huomenna yritän varmaan aamulla nopeasti vilkaista, keksinkö ratkaisua, ja jos en keksi, niin lähden todennäköisesti haastamaan helpointa ominaisuutta. Tyypillisesti helpoimmaksi luulemani ominaisuudet ovat toisaalta juuri niitä vaikeimpia. Olin lähes varma, että saisin tämän ominaisuuden toimimaan, joten en voi sanoa olleeni päivään erityisen tyytyväinen.

Torstai 15.6.2018

Kuten eilisessä kirjoituksessa mainitsinkin, niin eilisen, tämän ja seuraavan päivän tavoitteet ovat samat eli siis kolme isompaa kokonaisuutta. Tänään itse asiassa selvisikin, että yksi isommista kokonaisuuksista olikin oikeastaan aika pieni. Kyseessä oli kokonaisuus, jonka olin arvioinutkin helpoimmaksi. Tein sen tänään ja lähinnä rupesi epäilyttämään se, menikö tämä nyt liiankin hyvin. Testasin ominaisuuden päällisin puolin ja se näytti toimivan, eli otetaan nyt sitten voitto tästä kotiin.

Rupesin sitten korjaamaan eilen ilmaantunutta bugia toiseksi vaikeimmassa ominaisuudessa ja sainkin sen korjatuksi, mutta samalla ilmeni uusi bugi. Tai oikeastaan kyseinen bugi oli ollut olemassa eilenkin, mutta en ollut huomannut sitä. Tarkemmin sanottuna tässä ei ole varsinaisesti edes kyse bugista, vaan erään tähän ominaisuuteen liittyvän koodin toimintatavasta, jonka joudun jotenkin ratkaisemaan tai kiertämään. Minulla on teoria, miten se voisi onnistua, mutta yritän sitä vasta huomenna, sillä tänään aika ei riittänyt.

Halusin myös aloittaa vaikeimmaksi arvioimaani ominaisuutta ja sain sen tavallaan myös tehdyksi. Se ei toimi täysin speksien mukaisesti, mutta idea ja runko toimivat aivan kuten sovittu. Se, miksen saanut toimimaan sitä täysin speksien mukaisesti, ei johdu bugista, vaan enemmänkin siitä, että en tiedä miten erään tähän liittyvän koodinpätkän voisi toteuttaa oikealla tavalla. Olen kuitenkin jokseenkin tyytyväinen, että sain edes tämän ratkaisun, koska olen aikaisemminkin yrittänyt tehdä tätä samaa ominaisuutta siinä onnistumatta. Olen päivään tyytyväinen, sillä kaikki kolme ominaisuutta ovat loppusuoralla tai tehtyinä.

Perjantai 15.6.2018

Tänään viimeistelin keskiviikkona ja torstaina tehtyjä ominaisuuksia jo keskiviikkona asettamani tavoitteen mukaisesti. Huomasin myös pari pienempää bugia, joita en ehtinyt täysin korjata tänään. Lisäksi tähän päivään sisältyi palaveri. Totta puhuakseni päivä ei ollut erityisen tuottoisa koodin osalta, mutta ei kai jokaisen päivän tarvitsekaan olla. Saavutin keskiviikkona asetetut tavoitteet odotusten mukaisesti, eli sain tehtyä sovellukset noin 80-prosenttisesti. Olisin tietysti toivonut, että olisin saanut tehtyä kaikki täydellisesti, mutta oikeastihan viimeiset 20-prosenttia ovat hitaimmat prosentit ainakin kokemusteni mukaan.

Viikkoanalyysi

Viikko oli aika tyypillinen. Asetetut tavoitteet saavutettiin odotusten mukaisesti, mutta mitään itsensä ylittämistä ei tapahtunut, vaikka varmasti kehityin ohjelmoijana tälläkin viikolla. Huomasinkin, että olen kehittynyt paljon varsinkin bugien syiden arvioimisessa. En ole varma, johtuuko tämä siitä, että tunnen ohjelmiston nyt paremmin kuin alussa, vai ovatko taidot tällä osa-alueella vain kasvaneet. Tietysti kyse voi olla molemmista. Viikko on tietysti lyhyt aika, ja ensi viikolla tilanne saattaa olla aivan toinen. Oppimiseni on tapahtunut suurimmaksi osaksi tekemisen kautta, sillä en ole lukenut teoriaa niin paljon viime aikoina, vaikka tarkoitus on kyllä ollut. Itse asiassa katson dokumentaatioitakin vähenevissä määrin, mistä en tiedä, onko se hyvä vai huono asia.

Viime viikolla oli puhetta parhaista käytänteistä. Olen pyrkinyt noudattamaan parhaita käytänteitä, ja itse asiassa huomasin tässä viikolla, että käytänkin linteriä, vaikken edes ollut ajatellut asiaa. Kyseinen linter tuli ohjelmistoympäristöni mukana automaattisesti. Linter on siis eräänlainen työkalu, joka auttaa löytämään bugien lisäksi tyylivirheitä.

Nyt miettiessäni, mikä on osa-alue, jossa minulla olisi eniten parannettavaa, niin mieleeni tulee heti testaus. Testaan koodin yleensä pintapuolisesti, mutta se ei riitä, sillä turhan monesti jo valmiiksi luulemani koodi sisältääkin jonkin bugin, jota saa sitten korjailla. Bugien korjaus on yleensä aina vaikeampaa, mitä kauemmin se on sovelluksessa ollut. Pahimmillaan bugista tulee ominaisuus, mikä tarkoittaa sitä, että jotain tiettyä bugia ei välttämättä ole edes mahdollista korjata.

Pyrin tulevalla viikolla tekemään enemmän testausta ja merkkaan testitapaukset esimerkiksi Exceliin. Excel ei ole mikään paras ohjelma varmastikaan, mutta uskoisin, että se ajaa asiansa.

Käytimme Exceliä myös ammattikorkeakoulussa aikanaan apuna. Tässä täytyy kuitenkin muistaa, että aika on rajallista ja testausta voisi tehdä kokoaikatyönä, mutta olen ohjelmistokehittäjä, ja testaus on siinä mielessä toissijainen homma. Tavoitteena on siis tehdä aivan perusmuotoista testausta, jota olen välillä tehnytkin, mutta en ole esimerkiksi dokumentoinut sitä erityisen hyvin.

Sonmez kirjoittaa blogissaan, että ohjelmistokehittäjät harvoin oikeasti tietävät mitään testauksesta. Monesti ohjelmistokehittäjät haluavat vain nopeasti julkaista sovelluksen ja testaushan vain hidastaa sitä. Testaus on kuitenkin yksi tärkeä osa ohjelmistokehittäjän taitoja. Sonmez kertoo myös, että testaus on ennen kaikkea riskien vähentämistä. Paino on tässä sanalla vähentäminen, koska mistään ohjelmistosta ei saa oikeasti bugitonta, eikä täydellistä testausta voi suorittaa. Ohjelma on siis testattava niin, että todennäköisyys mahdollisille ongelmille on testauksen jälkeen mahdollisimman pieni. (Sonmez 2017, viitattu 17.6.2018.)

3.7 VIIKKO 25

Maanantai 18.6.2018

Päivän tavoitteena oli saada tiettyä tietoa näkymään halututussa paikassa. Kyseinen tehtävä on yllättävän haastava. Päivä lähti liikenteeseen aika hitaasti, ja aloin tuntemaan itseni aamusta hieman sairaaksi, joten otin välillä nokosia ja jatkoin nokosten jälkeen töitä. Tätä jatkui iltaan asti, jotta saan tunnit täyteen. Jälkeenpäin ajateltuna minun olisi varmaan pitänyt suosiolla luovuttaa ja kerätä voimia seuraavalle päivälle, sillä tästä päivästä ei oikein tullut lopulta mitään. Sain kuitenkin jotain hahmotelmia aikaiseksi, miten voisin toteuttaa kyseisen työn, ja jatkan tästä sitten huomenna.

Tiistai 19.6.2018

Tänään minulla oli nuhakuume, jota eilinen päivä oikeastaan myös enteili. Alkuperäisenä tavoitteena olisi ollut osallistua palaveriin ja saada eilinen aloittamani tiedon näyttäminen valmiiksi. Palaverin jouduin jättämään väliin, sillä en kuumeessa voinut lähteä pyöräilemään.

Pyrin kuitenkin saamaan valmiiksi sen, mitä eilen jätin kesken. Aamun nukuin, mutta päivällä olin kerännyt tarpeeksi voimia, että pystyin jälleen koodaamaan. Myös kuume oli laskenut. Tein päivän sekä iltapäivän töitä ja onnistuin tässä datan näyttämisessä. Joudun huomenna varmaan vielä käymään koodin kunnolla läpi, sillä kuumeessa tehty koodi saattaa olla jossain määrin väärin. Tekemäni koodin idea vaikuttaa kuitenkin oikealta, mihin olen varsin tyytyväinen. Olosuhteisiin nähden tämä päivä oli lopulta ihan hyvä.

Keskiviikko 20.6.2018

Tänään tavoitteena oli viimeistellä ja käydä läpi edellisen kahden päivän aikana tehtyä koodia. Tavoitteena oli myös tehdä viime perjantaina kesken jäännyyttä koodia loppuun, sillä siihen oli jäänyt bugeja. En voinut tänäänkään erityisen hyvin, mutta vointi oli silti paranemaan päin. Kuumetta ei ollut. Aamupäivän työskentelin edellisellä viikolla tehdyn koodin parissa. Työskentely sujui kuitenkin hieman tahmaisesti. Sain jotain pientä aikaan, mutta ajatus ei kulkenut tarpeeksi, jotta olisin saavuttanut tavoitteeni tältä osalta. Päätinkin siis vaihtaa tehtävää ja katsastaa eilisen ja maanantain tuotoksia.

Tein parannuksia eilisen ja maanantain koodiin. Kyseinen työ sujui yllättävän hyvin, eikä suurempia ongelmia ilmaantunut. Myönnän kuitenkin, että kyseinen koodi ei ollut siisteintä koodiani, mutta siistin sen joskus myöhemmin paremmalla ajalla. Aikaa jäi myös muutaman bugin korjaukseen. Kyseessä oli todella pieniä bugeja, eikä niistä oikeastaan ole suurempaa kerrottavaa. Käytännössä tällaiset pienet bugit korjautuvat monesti vain arvoa x tai y muuttamalla.

Torstai 21.6.2018

Tänään tavoitteena oli tehdä sitä missä eilen epäonnistuin, eli viimeistellä viime perjantain koodiani. Eilen työ ei sujunut, mutta tänään se sujui. Olokin on jo parempi, vaikka yleistila ei vieläkään ole normaali. Ongelmana oli uusi syntaksi, johon en ollut ennen törmännyt. Eilen ilmeisesti

oletin tietäväni, miten kyseinen syntaksi toimii, mutta tänään sitten oikeasti selvitin asian. Aloitin myös uutta ominaisuutta, mutta se jäi vielä kesken aika pahasti. Tein myös mielenkiintoisen huomion, siitä miten paljon terveydentila vaikuttaa koodin ymmärtämiseen. Eilen moni asia, mikä tänään oli täysin selvää, näytti heprealta.

Viikkoanalyysi

Mainitaan heti aluksi, että perjantain päiväraportti puuttuu tarkoituksella, sillä olin silloin juhannuksen vietossa. Tämä viikko ei ollut erityisen tuottoisa johtuen nuhakuumeesta, johon sairastuin. Yleiskuntoni oli laskenut, ja se näkyi myös ohjelmoinnissa. Jos ohjelmointi olisi vähänkään fyysisempää, niin en olisi edes yrittänyt tehdä töitä. Nyt tein töitä vointini mukaan, eli yleensä tunnin tai kahden jaksoissa, minkä jälkeen seurasi tunti lepoa ja sitten sama uudestaan. Kyseinen järjestely onnistui, koska tein etätöitä. Fyysiset palaverit jouduin kuitenkin omalta osaltani perumaan, mutta niitä ei olisi ollutkaan onneksi kuin tiistaina.

Viime viikolla otin tavoitteekseni kehittää testaustaitoja tämän viikon aikana. Sairauden ja muiden olosuhteiden vuoksi en kuitenkaan erityisemmin jaksanut keskittyä testaamiseen, vaan käytin vähäisen energian suhteellisen perustyön tekemiseen. Siirränkin siis kyseisen tavoitteen ensi viikolle, johon se sopii hyvin, sillä mielestäni tämän viikon koodi on testauksen tarpeessa.

Tällä viikolla kehittyivät eniten suunnittelutaidot. Piirtelin aika paljon hahmotelmia mahdollisista ratkaisuista vihkoon, samalla kun levähdin, vaikken tästä päiväraporteissa maininnutkaan. Nämä vihkoon piirretyt ratkaisut toimivat siten, että niiden avulla pääsi yleensä alkuun ja ne auttoivat keskittymään itse tekemiseen. Osittain kyseisen paperilla suunnittelun avulla sain tällä viikolla kuitenkin ihan hyvin asioita aikaiseksi. Suunnittelen asioita paperille myös terveenä, mutta niitä ei tule ehkä mietittyä niin syvällisesti. Nyt oli hieman pakko miettiä sitä, mitä laittoi paperille, koska kuume vei tietokoneelta sänkyyn. Joskus voikin olla selvästi hyvä pysähtyä ja miettiä hieman, mitä sitä ollaankaan tekemässä ja miksi.

Koska tämän viikon koodi pitäisi kuitenkin vielä käydä läpi varmuuden vuoksi, niin apuna voisi käyttää myös jotain uutta metodia, jo mainitun testauksen lisäksi. Olen pitkään halunnut jo käyttää sellaista metodia kuin kumiankka debuggaus (rubber duck debugging). Yksinkertaisuudessaan siinä selitetään koodi tai mahdollinen ongelma kumiankalle, jolloin huomaa helposti, osaako asiaa oikeasti selittää. Kumiankka on tyypillisesti aivan perinteinen kumiankka, joita saatetaan

käyttää myös vaikka ammeessa. Mikäli asiaa ei osaa selittää, niin koodissa voi olla parantamisen varaa. Samalla myös huomaa helposti mahdollisia virheitä, joita ei muuten välttämättä tule ajatelleeksi. Selittäminen auttaa muistamaan unohtuneita asioita tai keskittymään itse ongelman ytimeen. Kumiankka-metodi voi kuulostaa aluksi hieman joltain vitsiltä, mutta se on oikeasti aika laajasti käytetty metodi ja terminä tunnettu. Kumiankka ei sovellu tietenkään ainoaksi debuggaus keinoksi, mutta se on hyvä lisä, jos perinteiset keinot eivät toimi. (Coleman 2018, viitattu 24.6.2018.)

3.8 VIIKKO 26

Maanantai 25.6.2018

Päivän tehtävänä on toteuttaa eräänlainen navigaation tyyppinen ratkaisu, jota nyt kutsun selkeyden vuoksi vain navigaatioksi. Kyseinen navigaatio toteutetaan jo olemassa olevaan koodiin, joten se tuo omia haasteita lisää. On hyvinkin mahdollista, että toteutus kestää tässä huomiseen asti.

Tehtävän toteutus lähti hyvin käyntiin ja sain navigaation idean toimimaan varsin nopeasti selaimen konsolissa. Konsoli onkin ehkä tärkein työkalu mitä olen edellisten viikkojen aikana käyttänyt. Jostain syystä navigaatio ei kuitenkaan toiminut täysin halutulla tavalla selaimessa itsessään. En löytänyt syytä navigaation toimimattomuudelle ja päätin siirtää sen selvittelyn seuraavaan päivään.

Tiistai 26.6.2018

Tämän päivän tavoitteena oli viedä eilen aloitettu navigaatio loppuun, korjata muutama bugi, luoda pohjaa tuleville lisätoteutuksille sekä testata tehtyä koodia. Navigaatio ei jostain syystä toiminut selaimessa ja aloitin heti aamusta etsimään syitä. Bugin etsimiseen meni puoli päivää, ja lopulta syyksi selvisi yksi muuttuja, jossa tieto ei käyttäytynyt halutulla tavalla. Itse bugin korjaus vei muutaman sekunnin. Periaatteessa olen pettynyt siihen, että en löytänyt näin yksinkertaista bugia nopeammin, toisaalta pahimmat bugit ovat yleensä varsin pieniä.

Tehtävänä oli myös korjata muutama muu bugi, joissa syy oli jo enemmän tai vähemmän selvillä. Uskoakseni kyseiset bugit korjautuivat, ja testasin ne myös läpi. Testauksessa käytin Exceliä. Päivä oli kuitenkin tässä vaiheessa jo aika lopussa, ja priorisoin tulevan pohjan lisätoteutuksille korkeammalle kuin syvemmän testauksen toteuttamisen. Kuten olen viikkoraporteissanikin mainninnut, niin testaamiseen saisi helposti kulumaan koko päivän. Minulla ei kuitenkaan ole sellaista aikaa, joten joudun tekemään kompromisseja. Toteutin pohjan lisätoteutuksille ja jatkan tästä ensi viikolla.

Keskiviikko 27.6.2018

Tänään tavoitteena oli tehdä useampia eri ominaisuuksia työn alla olevaan ohjelmaan. Mainitaan selkeyden vuoksi, että nämä ominaisuudet eivät liity alkuviikon tekemisiin. Ominaisuuksien yksityiskohdista en voi puhua tarkemmin, mutta ne olivat varsin pieniä ohjelman kokonaiskuvaan nähden. Työtä kyseisissä ominaisuuksissa kuitenkin riittää, sillä niitä tehdessä täytyy erityisesti ottaa huomioon jo olemassa oleva koodi.

Onnistuin tekemään kaksi tai kolme ominaisuutta, hieman näkökulmasta riippuen. Olen tähän kohtalaisen tyytyväinen. Suurempia ongelmia ei ilmennyt. Aika kului lähinnä toteutustavan suunnitteluun. Pienemmällä suunnittelun määrällä olisin varmasti voinut tehdä tuplasti enemmän ominaisuuksia, mutta laatu on aina määrää tärkeämpää.

Torstai 28.6.2018

Tänään halusin viedä eilen tehtyjä ominaisuuksia hieman eteenpäin, vaikka itse ominaisuudet olivatkin jo sinällään valmiina. Vein ominaisuuksia eteenpäin ja sain niistä mielestäni paremmat kuin aikaisemmat versiot olivat. Yksinkertaistin myös hieman koodia. Lisäksi kävin läpi ohjelmaan liittyvää dokumentaatiota sekä työn alla olevaa ohjelmaa, jos löytäisin jotain jatkokehitettävää. Jatkokehitykseen keksin muutaman idean, mutta joudun hieman konsultoimaan muita henkilöitä niihin liittyen. Päivään kuului myös palaveri ja hieman pilvipalveluiden käyttöä. Pilvipalvelu ei toiminut aivan kuten odotin, joten käyn huomenna sitä enemmän ajatuksen kanssa läpi.

Perjantai 29.6.2018

Tänään tavoitteena oli viimeistellä ja testata eilen ja keskiviikkona tehtyä koodia, saada käyttämästämme pilvipalvelusta parempi kuva ja mahdollisesti aloitella jotain uutta isompaa ominaisuutta. Tein koodin viimeistelyn heti aamusta ja iltapäivästä tutustuin pilvipalveluun tarkemmin. Joudun jälleen jättämään hieman yksityiskohtia pois, mutta siellä törmäsin kuitenkin ongelmaan, johon en keksi ratkaisua. Kaiken pitäisi olla kunnossa, mutta ei ole. Epäilen, että kyseessä on jokin oikeuksiin liittyvä ongelma. Laitoin sähköpostiviestiä eteenpäin ja katson sitten, että miten asiassa edetään. Aloitin myös erään isomman ominaisuuden teon, mutta se jäi todella pahasti puolitiehen, enkä osaa edes sanoa vielä, olenko oikeilla jäljilläkään.

Viikkoanalyysi

Viikko oli aika tyypillinen, mutta panostin tällä viikolla enemmän testaukseen. Testaus oli kuitenkin toissijaista, sillä en ole kuitenkaan varsinainen testaaja. Testaus kuuluu kuitenkin myös ohjelmistokehittäjän työkalupakkiin, kuten viime viikolla viikkoanalyysissäkin oli puhetta. Uutena asiana oli pilvipalvelut, joihin tutustuin pintapuolisesti. En saanut haluttuja asioita toimimaan, mutta näitä ongelmia selvitän tarkemmin ensi viikolla, jolloin voin kysyä myös neuvoa kohtaamiini ongelmiini.

Ongelmat, joita nykyään kohtaan, liittyvät vähemmän ja vähemmän siihen, pystyykö tai osaako jotain tiettyä asiaa koodata. Välillä joutuu tekemään kompromisseja, mutta se kuuluu työnkuvaan. Nykyään ongelmat liittyvät enemmän alustoihin, tekniikkoihin tai ovat jonkinlaisia logiikkavirheitä, joita joutuu sitten jäljittämään. Olen kehitykseeni kuitenkin varsin tyytyväinen, vaikka mitä enemmän ohjelmointia tekee, niin samalla ymmärtää kuinka paljon on vielä opittavaa.

Tavoitteenani on saada ohjelmoinnista mahdollisimman kattava kuva nyt, kun olen vielä ammattikorkeakoulussa oppimassa. Oikeastaanhan ohjelmointi on logiikkaa ja siksi looginen ajattelu on erittäin tärkeä ominaisuus ohjelmoijalle. Siksi liityinkin Reaktorin ja Helsingin yliopiston järjestämälle ilmaiselle tekoälyaiheiselle kurssille. Tarkalleen ottaen liityin kurssille jo alkukesästä, mutta rupesin vasta nyt tutkimaan sitä tarkemmin. Olen kuitenkin kokenut, että siitä on ollut paljon apua loogiseen ajatteluun ja siten se on myös auttanut työelämässä.

Kurssin nimi on Elements of AI. Se on kahden opintopisteen laajuinen ja olen nyt siinä noin puolessa välissä. Kurssin tavoitteena on poistaa salaperäisyyttä tekoälyn ympäriltä. Se on suunnattu kaikille taustasta riippumatta. (University of Helsinki & Reaktor 2018, viitattu 1.7.2018.)

Käyn Elements of AI-kurssia vapaa-ajallani omaehtoisesti. Kurssilla ei sinällään ohjelmoida mitään, mutta käydään läpi esimerkiksi todennäköisyyksiä siitä, mitä tekoäly oikeasti on, miten tekoäly toimii ja miten se oppii uusia asioita. Vaikka kurssi onkin peruskurssi, niin siitä on oikeasti oppinut paljon ja uskon myös ohjelmointitaitojeni kasvaneen hieman kurssin myötä. Pyrin käymään kurssin nyt seuraavan parin viikon aikana läpi. Tekoälykurssi sai minut myös ymmärtämään paremmin sen, mihin matematiikkaa ja fysiikkaa voi soveltaa ohjelmoinnissa.

Koska olen nyt tehnyt lähinnä perus-front-end -ohjelmointikehitystä, niin en ole oikeastaan perusmatematiikkaa kummempaa joutunut missään käyttämään. Matematiikka ja fysiikka on uskoakseni kuitenkin seuraava askel, jossa joudun parantamaan taitojani, vaikka monesti matematiikka ja fysiikka ovatkin enemmän sidoksissa back-end-puoleen. Ammattikorkeakoulussakin on ollut toki matematiikkaa, mutta se on ollut hieman käytännönläheisempää, eli matematiikkaa, johon törmää nimenomaan front-end puolella. Itse haluan kuitenkin tulevaisuudessa olla full-stack -ohjelmistokehittäjä, mikä tarkoittaa sitä, että osaa front-end ja back-end-puolet. Vaikken päätyisikään full-stack-ohjelmistokehittäjäksi, niin paremmat matemaattiset valmiudet auttavat varmasti loogisessa ajattelussa. Todennäköisesti tutustun nyt aluksi enemmän pintapuolisesti matemaattiseen puoleen ja myöhemmin jatkokouluttaudun YAMK:n tai yliopiston kautta lisää. Tällä hetkellä pääpaino on kuitenkin töissä eli front-end -puolella ja siinä kehittymisessä, joten miksikään matemaattiseksi neroksi ei ole tarvetta kehittyä yhdessä yössä.

3.9 VIIKKO 27

Maanantai 2.7.2018

Tavoitteena oli jälleen toteuttaa uutta toiminallisuutta ohjelmistoon. Toiminallisuudet, joita toteutan, ovat itsessään suhteellisen pieniä, millä tarkoitan, että toteutuksessa menee muutama tunti tai korkeintaan päivä.

Päivä lähti varsin hyvin käyntiin, ja sain useamman toisiinsa liittyvän ominaisuuden tehtyä prototyyppiasteelle. Kyseessä oli siis oikeastaan eräänlainen rypäs ominaisuuksia. Toiminallisuuksiin jäi kuitenkin muutama bugi, joiden syyn jo tiedän. Korjaan ne huomenna, ja tässä onkin huomionarvoista se, että tässä kunnossa koodia ei saisi laittaa versionhallintaan, sillä jos joku muu jatkaisi tästä niin hän ei tietäisi kyseisistä bugeista ja saattaisi yllättyä myöhemmässä vaiheessa.

Tiistai 3.7.2018

Päivän tavoitteet liittyivät eilisen koodin saattamisen siihen kuntoon, että sen kehtaa laittaa versionhallintaan. Lisäksi ajattelin korjata muutaman pienemmän bugin. Päivän tehtävät ovat siis aika tyypillisiä.

Myös päivä oli lopulta tyypillinen, ja kaikki meni noin suunnilleen kuten suunniteltu. Sain tehtyä kaiken, mitä ajattelinkin, ja lisäksi selvitin viime viikon ongelmat pilvipalveluun liittyen. Ratkaisu oli jälleen aika yksinkertainen, ja löysin senkin vahingossa törmätessäni toiseen pienempään ongelmaan. Vaikka päivä olikin varsin normaali, niin se oli aika opettavainen kaikkine pienine ongelmineen, joita ratkoin tänään.

Keskiviikko 4.7.2018

Tänään aloitin eilisestä poiketen isomman ominaisuuden tekemisen, jonka toteuttamiseen käytännössä menee uskoakseni muutama päivä. Tavoitteena olikin tänään päästä hyvään alkuun ja tehdä hyvä pohja seuraavia päiviä varten. Käytännössä tein tänään useampiakin eri pohjia, ja mietin niiden hyötyjä ja haittoja. Kaikki pohjat olivat sinällään toteutettavissa. Päädyin lopulta ratkaisuun, joka on vaikeampi toteuttaa tässä vaiheessa mutta helpottaa todennäköisesti myöhempää työtä. Käytin myös minulle uusia ohjelmointikirjastoja tänään, joiden sisältöön en kuitenkaan syvenny tarkemmin johtuen salassapidollisista syistä. Päivä sujui varsin hyvin ja opin paljon uutta, varsinkin kun käytössä oli täysin uusi kirjastokin. Nykyään uuden kirjaston oppimiseen ei kuitenkaan mene kovinkaan kauan, vaan pelkkien dokumentaatioiden katsominen riittää jo aika pitkälle.

Torstai 5.7.2018

Päivän tavoitteenani oli viedä eilen aloitettu ominaisuus siihen pisteeseen, että voisin viimeistellä sen perjantaina. Lisäksi tiedossa oli palaveri.

Päivä sujui kokonaisuutena varsin hyvin. Päivän alussa tein eilisen koodin pohjalta version ominaisuudesta, jossa oli vielä aika paljon rajoitteita, mikä olisi ollut jo itsessään hyvä saavutus koko päivälle. Koska päivä oli kuitenkin vasta alussa, päätin poistaa pahimmat rajoitteet ja sain varsin toimivan version siitä. Tässä vaiheessa riitti lyhyt testaus, koska kyseessä olevaan ominaisuuteen tulee todennäköisesti vielä paljon isojakin muutoksia.

Palaveri oli aika tyypillinen. Omalta osaltani kerroin tekemiseni ja sen mitä teen jatkossa. Päivään voi kokonaisuutena olla erittäin tyytyväinen.

Perjantai 6.7.2018

Päivän tärkein asia oli mielestäni palaveri. Tavoitteenani palaverista oli saada uutta tietoa siitä, että miten jatkaa ohjelmiston kehittämistä. Valmistauduinkin heti aamusta palaveriin käymällä edellisten päivien koodin läpi ja laittamalla sen sitten versionhallintaan.

Palaveri sujui varsin hyvin. Palaverin ansioista tiedän nyt, mitä seuraavat viikot tuovat tullessaan. Aloitinkin jo hieman seuraavan viikon töihin liittyvää suunnittelua. Tämä päivä oli siis kokonaisuutena ajatellen enemmänkin töiden suunnittelua kuin niiden varsinaista tekemistä. Olen päivään tyytyväinen, sillä käsitykseni nyt seuraavan viikon töistä kirkastui.

Viikkoanalyysi

Kuten viime viikko niin myös tämä viikko oli varsin tyypillinen. Osaamiseni kehittyminen on tietyllä tapaa tasaantunut, mutta se ei ole mielestäni huono asia. Painotan vielä sitä, että tasaantumisella en tarkoita pysähtymistä. Alkuviikkoina tuli kuitenkin paljon täysin uutta asiaa, jolloin jokainen päivä ja siten myös viikko oli varsin erilainen toisistaan. Tällä hetkellä toistan paljon asioita, joita

vielä opettelin ensimmäisillä viikkoina, varsinkin jos kyse on ReactJS:istä. Toistojen kautta oppimani asiat jäävät sitten lopulta syvemmin mieleen.

Kirjoitin viime viikolla myös hieman matematiikasta, fysiikasta ja tekoälykurssista, jota käyn. Tarkoituksena oli tällä viikolla tehdä tekoälykurssia eteenpäin ja hieman sitä kautta tutustua matematiikkaan ja fysiikkaan, jolloin myös loogisen ajattelukykyni pitäisi kehittyä entisestään. Tämä kaikki tapahtuu siis vapaa-ajalla, sillä en suoraan tarvitse edellä mainittuja taitoja työssäni, vaan hyöty tulee paremmasta loogisesta ajattelusta. En kuitenkaan ehtinyt toteuttaa tätä suunnitelmaa, sillä kalenteri oli aika tiiviisti varattu. Ensi viikolla aikaa pitäisi kuitenkin olla enemmän, jolloin pystynen tekemään tekoälykurssia eteenpäin.

Huomasin, että en ollut ottanut puheeksi keskittymistä viikkoraporteissani, vaikka se on varsin tärkeä asia työni kannalta. Keskittyminen on kuitenkin aika olennaista, jotta koodia saa aikaiseksi. Jos koodia yrittää pakottaa ilman keskittymistä, niin siitä ei tule ainakaan omalta osaltani mitään. Myönnän, että keskittymisen suhteen minulla voisi olla parannettavaa, sillä välillä idea siitä, mitä olin tekemässä, karkaa. Monesti lähden ajattelemaan koodia liian pitkälle tulevaisuuteen, jolloin sen hetkinen tilanne unohtuu. Keskittymistä käytiin eri näkökulmista läpi myös ammattikorkeakoulussa ainakin Opiskelijana ammattikorkeakoulussa, Käyttäjäkokemus ja käytettävyys, sekä Käyttäjäkoulutus ja neuvonta -kursseilla. Voisinkin katsoa jokin päivä edellä mainittujen kurssien materiaaleja hieman läpi. Jonkinlaisen suunnitelman tekeminenkään keskittymisen osalta ei varmaan haittaisi.

Keskittyminen on kaikilla varmasti yksilöllistä. Toinen tykkää kuunnella musiikkia ja joku toinen taas pitää hiljaisesta tilasta. Keskittymisen parantamiseen löytää netistä kuitenkin varsin helposti vinkkejä, joista kannattaa valita itselle parhaiten sopivimmat neuvot. Tartakovsky neuvoo artikkelissaan esimerkiksi menemään ulos, tekemään to-do-listoja, etsimään oikean stimulaatiotason (ei liian tylsää, ei liian vaikeaa) sekä reflektoimaan tekemisiään (Tartakovsky 2016, viitattu 8.7.2018.)

Otan ensi viikon tavoitteeksi ajatella asioita enemmän psykologian kautta ja tehdä jotain konkreettista myös asian eteen, kuten laatia jonkinlaisen käytännön esimerkiksi juuri keskittymisen

osalta. Käytännön laatimisella tarkoitan itselleni oikeiden keskittymismetodien etsimistä ja soveltamista. Mainitaan vielä loppuun, että vaikka puhuinkin paljon keskittymisestä, niin se ei ole mitenkään huonolla tasolla nytkään, mutta näen siinä silti parantamisen varaa. Keskittymisen lisäksi on varmasti tärkeää keskittyä esimerkiksi vireystilaan, töistä palautumiseen, oikeanlaiseen ravintoon ja terveyteen ylipäätänsä. Uskoisin, että kaikki edellä mainitut tekijät tukevat toisiaan.

3.10 VIIKKO 28

Maanantai 9.7.2018

Tänään kävin koodia läpi bugien varalta ja pyrin tekemään samalla samasta koodista parempaa, johon myös päivän tavoitteet liittyivät. Koodissa ei sinällään ollut mitään vikaa, vaan kyseessä oli tyypillinen koodin läpikäyminen. Kävin lähinnä omaa koodiani läpi, sillä tunnen sen parhaiten. Mahdolliset ongelmat on hyvä huomata, ennen kuin ne ovat ongelmia. Mitään isompaa ongelmaa ei löytynyt, mutta tein silti pieniä korjauksia koodiin. Mielestäni päivän suurin oppi oli kuitenkin se, että koodin toiminta tuli jälleen tutuksi, sillä se oli hieman jo päässyt unohtumaan. Kokonaisuutena ajatellen päivä oli varsin onnistunut.

Tiistai 10.7.2018

Tänään tavoitteena oli testata, että kaikki toimii työn alla olevassa ohjelmistossa kuten pitää ja korjata mahdolliset viat. Kaikki toimi yllättävänkin hyvin, joten en joutunut tekemään suurempia korjauksia. Koska aikaa jäi hyvin yli, tein pari prototyyppiä mahdollisesti tulevaisuudessa lisättävistä ominaisuuksista. Prototyyppien ideana on esitellä sitä, miten lähtisin toteuttamaan tiettyä asiaa.

Päivä oli kuitenkin myös tänään varsin hyvä. Työtä oli vähemmän kuin odotin, sillä testauksessa ei ilmennyt suurempia ongelmia. Sain siis tehtyä kaiken, mitä aluksi ajattelin, ja hieman enemmänkin.

Keskiviikko 11.7.2018

Tänään tarkoituksena oli aloittaa isompi kokonaisuus ohjelmistoon. Kyseessä on siis yksittäinen kokonaisuus, jota on tarkoitus tehdä loppuviikko ja mahdollisesti hioa myöhemmin. Aloitin työnteon tänään hieman normaalia myöhemmin, mutta etätyössä on se hyvä puoli, että se onnistuu helposti. Aloitin siis kello 12 kun, normaalisti teen töitä klo 8–16 välisenä aikana. Kello 12–20 oli oikeastaan aika hyvä aika tehdä töitä.

Työt sujuivat hyvin, ja sain tekemäni kokonaisuuden hyvin alkuun. Päivän ongelmat liittyivät lähinnä suunnittelupuoleen eli siihen, että miten lähden koodia tekemään. Myös ohjelmistoympäristössä oli pieniä ongelmia, jotka ratkesivat asetuksia muokkaamalla.

Torstai 12.7.2018

Tänään jatkoin siitä, mihin eilen jäin. Tavoitteena oli tehdä eilen aloitettuani kokonaisuutta mahdollisimman pitkälle mielellään siihen kuntoon, että sitä voisi jo esitellä.

Käytännössä pääsin siihen pisteeseen, että koodini oli kyllä esittelykunnossa, muttei kuitenkaan kaikilta osin valmis. Joudun toki hieman vielä hiomaan koodia siistimmäksi, jotta sen kehtaa laittaa myös versionhallintaan. Koodin hiominen on kuitenkin huomisen asia. Syy siihen, miksen saanut koodia valmiiksi, ei johtunut sinällään suurimmista ongelmista vaan siitä, että muutaman kohdan tekeminen osoittautui yllättävän työlääksi. En ole pettynyt päivään, mutta en ole kyllä varsin tyytyväinenkään siihen.

Perjantai 13.7.2018

Päivän tavoitteena oli viimeistellä eilisen koodia ja laittaa se versionhallintaan. Lisäksi tiedossa oli palaveri.

Tässä päivässä ei ollut mitään yllätyksiä. Päivä oli ehkä normaalia päivää rauhallisempi, sillä en tehnyt tai aloittanut mitään suurempaa. Sain koodin viimeisteltyä ja laitettua sen versionhallintaan. Palaverin jälkeen tein vielä pieniä muutoksia ja laitoin koodin uudestaan versionhallintaan. Loppupäivästä kävin koodia läpi ja suunnittelin hieman sitä, mitä lähden ensi viikolla tekemään.

Olisi tietysti ollut kiva, jos näin opinnäytetyöni viimeiselle päiväraportille olisi saanut paljon toimintaa, mutta kyllä tällainen rauhallinenkin lopetus sopii.

Viikkoanalyysi

Tällä viikolla teemana oli keskittyminen, josta olikin jo viime viikolla puhetta. En päiväraporteissani erityisemmin maininnut käyttämiäni metodeja, mutta yritin kuitenkin joka päivä etsiä parhainta mahdollista työskentelytapaa tulevaisuutta ajatellen. Keskittyminen ja työskentelytavat kuitenkin korreloivat koodin laatuun ja määrään.

Dan Goleman kirjoittaa artikkelissaan, että keskittyminen on eräänlainen henkinen lihas, sillä se vahvistuu siitä, mitä enemmän sitä käyttää. Golemanin mukaan keskittyminen myös korreloi menestymisen kanssa, riippumatta siitä mitä tekee. Tyypillinen ongelma on vaelteleva mieli. Golemanin artikkelissa annetaan vinkiksi vaeltelevaan mieleen, että keskity hengitykseen, minkä jälkeen yritä huomata, että mielesi on lähtenyt vaeltelemaan, sitten irtaudu ajatuksesta, johon mieli on vaeltanut ja, lopuksi pyri saamaan keskittyminen takaisin hengitykseesi ja pidä se siellä. (Goleman 2013, viitattu 3.9.2018.)

Erilaisilla netin keskustelufoorumeilla yleisesti suositeltiin menemään ulos, kirjastoon tai muuhun vastaavan paikkaan ohjelmoimaan. Tein näin alkuviikosta mutta lopetin kokeilun lyhyeen, sillä kannettavallani on varsin ikävä ohjelmoida. Kannettavan näyttö on mielestäni liian pieni koska olen tottunut käyttämään useampaa isoa näyttöä pöytäkoneella. Kokeilusta jäi kuitenkin se käteen, että jonkinlainen lenkki tai vaikkapa kauppareissu ohjelmoinnin ohessa voi olla ihan hyvä idea. Itselläni kauppareissu auttaa saamaan ajatukset pois ohjelmoinnista hetkeksi, jolloin takaisin tullessa aivot jaksavat keskittyä paremmin.

Keskiviikkona testasin sitä, kuinka hyvä idea olisi aloittaa työt hieman myöhemmin. Tulin loppupäätelmään, että se voi olla hyvä idea esimerkiksi kerran viikossa. Näin viikkorytmiin tulee enemmän vaihtelua ja esimerkiksi keskellä viikkoa saa nukkua pidempään. Aikaisemman vapaa-ajalta olevan kokemukseni perusteella joka päivä ei kannata kuitenkaan herätä kello 12, koska muuten siitä tulee nopeasti uusi normi. Otan kuitenkin varmastikin käyttöön sen, että herään kerran viikossa hieman myöhemmin. Normaalisti olen loppuviikosta hieman poikki unirytmini takia, mutta tällä viikolla sitä ongelmaa ei ollut.

Torstaina ja perjantaina testasin alkuviikon teoriaa siitä, toimiiko esimerkiksi lyhyt lenkkeily esimerkiksi noin tunnin välein. Mielestäni teoria toimi, sillä jaksoin keskittyä paljon paremmin. Toisaalta perjantain osalta lenkkeilyt tulivat hieman kuin itsestään, sillä päivään sisältyi palaveri, jonne menin ja josta tulin pyörällä. Ensi viikolla jatkan varmaan uusien parempien vastaavien toimintatapojen etsimistä.

Toinen tärkeä asia on tietenkin itse koodaukseen liittyvät toimintatavat, jotka ovat tietyllä tapaa jo rutinoituneet minulla. Uskallan väittää, että nykyiset toimintatapani ovat toimivat, sillä saan itsenäisesti toimivaa koodia aikaiseksi. Heikkouteni on mielestäni optimaalisten loogisten reittien löytäminen ongelman ratkaisemiseksi. Löydän yleensä ratkaisun ongelmaan kuin ongelmaan, mutta siinä menee aikaa ja ratkaisuja joutuu monesti vähän hiomaan jälkikäteen. Jotta ratkaisuista saa parempia, niin on tärkeää opiskella myös vapaa-ajalla, olipa opiskelemassa tai ei. Itse suoritin esimerkiksi tällä viikolla loppuun jo aikaisemmilla viikoilla mainitsemani Elements of AI kurssin. Tulen myös jatkossa käymään vastaavia avoimia kursseja, mikäli koen saavani niistä hyötyä ammattitaitoani ajatellen. Kurssien käyminen on erityisen tärkeää varsinkin, kun varsinaiset opiskelut loppuvat, sillä muuten helposti putoaa tekniikan kehityksen kyydistä. Asiaan vaikuttavat toki myös muut tekijät, kuten työn luonne.

4 POHDINTA

Opinnäytetyöprosessia aloittaessa osasin perusteet työtehtävistäni ammattikorkeakoulun opintojeni kautta ja osasin sen pohjalta toimia itsenäisesti jo varsin aikaisessa vaiheessa. Työvauhtini oli kuitenkin opinnäyteprosessin alussa varsin hidas, mutta se on kehittynyt huomattavasti näiden 10 viikon aikana. Opinnäyteprosessin alussa en myöskään ollut erityisemmin erikoistunut mihinkään kieleen vaan opetellut perusteita vähän kaikista kielistä, mikä on toki itsessään hyvä pohja. Olen käyttänyt opinnäyteprosessin aikana pääasiassa ReactJS:iä, jota osaankin nyt käyttää selvästi paremmin kuin muita aikaisemmin opiskelemiani ohjelmointikieliä. Aikaisemman kokemusteni perusteella ohjelmointikielten opiskelu kuitenkin tukee toisiaan. Oletan siis, että seuraavan vastaavan kielen opiskelu vie entistä vähemmän aikaa tälle nykyiselle tasolle.

Olen oppinut myös löytämään paremmin ratkaisumalleja erilaisiin ongelmiin opinnäyteprosessin aikana. Erityisesti uuden, vaikean, aikaisemmin kohtaamattoman ongelman kohdatessa en ala käyttämään liikaa aikaa ongelman ratkaisuun, jos selvää ratkaisua ei heti tule mieleen, vaan tällöin teen muita töitä ja yksinkertaisesti odotan sitä, että ratkaisu tulee itsestään mieleen. Yleensä ongelmaan kuin ongelmaan tulee jonkin ajan kuluttua jokin ratkaisu mieleen, ja se on myös tyypillisesti parempi kuin ratkaisu, jota olisin pakolla yrittänyt keksiä. Mielestäni on myös hyvä käytäntö osata asettaa jokaiselle päivälle jokin tavoite, kuten olen opinnäytteessäni tehnytkin. Konkreettinen tavoite auttaa paremmin analysoimaan päivää jälkeenpäin.

Psykologisten ratkaisumallien lisäksi olen myös oppinut luonnollisesti löytämään parempia teknisiä ratkaisuja. Tekniset ratkaisumallit perustuvat pitkälti siihen, mitä olen ratkaisut aikaisemmin. Kun ratkaisen tietyn ongelman, siitä jää mieleen skeema eli eräänlainen sisäinen toimintamalli, jonka avulla voi aina ratkaista vastaavan ongelman.

Osaan nykyään myös käyttää teoriaa paljon paremmin apuna erilaisten ratkaisumallien löytämiseen. Teorian lukeminen oli alkuvaiheessa toisinaan hieman vaikeaa, sillä kirjallisuudessa käytettiin paljon termejä, joita en täysin ymmärtänyt. Nyt olen kuitenkin oppinut ymmärtämään käytettyjä termejä, ja hieman monimutkaisempiakin selityksiä. Opinnäytetyön kirjoittamisprosessi on myös edesauttanut paljon teorian osaamista, sillä kirjoittaessa tekstiä on tärkeää ymmärtää mitä kirjoittaa.

Opinnäytetyön kirjoittamisprosessi on paremman teorian osaamisen lisäksi auttanut minua kehittymään kirjoittajana ja ajattelemaan kriittisemmin. Kirjoitustaidot tulevat tarpeeseen esimerkiksi dokumentoidessa omia töitä tai ylipäätänsä kommunikoidessa kirjallisesti, mikä tällä alalla on varsin yleistä. Kriittinen ajattelu on taas tärkeää entistä parempien toimintamallien löytämiseksi, sillä olen varma, etteivät nykyisetkään toimintamallini ole parhaat mahdolliset. Totta puhuen toimintamallini eivät varmaan koskaan ole parhaat mahdolliset, vaan tarkoitus on päästä mahdollisimman lähelle sitä. Kriittinen ajattelu on tärkeää myös tiedon etsinnässä, sillä esimerkiksi liian vanhat ratkaisut tiettyyn ongelmaan voivat olla alttiita tietoturvan kannalta. Myöskään ensimmäistä vastaantulevaa ratkaisua ei kannata ottaa vastaan, varsinkaan jos ei ymmärrä täysin mitä ratkaisussa tapahtuu.

Opinnäytetyön prosessin aikana huomasin myös, että avoimen lähdekoodin tarjonta on varsin laajaa ja, että avoimen lähdekoodin ohjelmissa on toisinaan isojakin puutteita, joihin pystyisin itsekin osallistua omalla osaamisellani. Avoimen lähdekoodin kirjastot ja sovellukset on toisinaan myös tehty turhan vaikeaksi käyttää, sillä kirjastoon tai sovellukseen on yritetty mahduttaa liikaa asiaa. Jos ja kun joskus on enemmän aikaa, tulen varmasti osallistumaan johonkin avoimen lähdekoodin projektiin. Olen myös itse miettinyt muutamaa omaa avoimen lähdekoodin projektia tai miksei myös suljetun lähdekoodin projektia, jolloin siitä voisi saada paremmin myös elannon tai sivutulon.

Olen hyödyntänyt työni analysointia erityisesti parempien toimintamallien löytämisen apuna kriittisen ajattelun lisäksi. Kirjoittaessa asiat tulee monesti tarkemmin mietittyä, jolloin huomaa helpommin puutteita omissa toimintamalleissa. Työn analysointi on myös auttanut löytämään suuntaa mahdollisille jatko-opinnoille, jotka ovat todennäköisesti enemmän matematiikan tai fysiikan parissa. Lisäksi työn analysointi varsinkin teorian kautta on auttanut paremmin ymmärtämään sen, mitä olen oikeasti ohjelmoidessani tekemässä.

Kokonaisuutena katsoen opinnäyteprosessi on ollut mielestäni varsin kehittävä kokemus, jossa olen voinut käyttää ja kehittää omaa asiantuntemustani. Tietojenkäsittelyn tutkinto on valmentanut varsin hyvin front-end-ohjelmistokehittäjän töihin, vaikka täytyy tietysti muistaa, että työtehtävät ja siten työntehtävien vaikeusaste voivat olla erilaisia yrityksestä riippuen. Tärkeintä on kuitenkin oma kiinnostus ja halu kehittyä omalla alallaan.

LÄHTEET

Coleman, A 2018. Rubber Duck Debugging: The best way to debug your code that you've never tried. Viitattu 24.6.2018, https://selftaughtcoders.com/rubber-duck-debugging/.

ESLint 2018. Rules. Viitattu 3.6.2018, https://eslint.org/docs/rules/.

Getify 2015. You-Dont-Know-JS. Viitattu 10.6.2018, https://github.com/getify/You-Dont-Know-JS/.

Goleman, D 2013. The Four Basic Moves to Strengthen Focus. Viitattu 3.9.2018, https://www.psychologytoday.com/us/blog/the-brain-and-emotional-intelligence/201310/the-four-basic-moves-strengthen-focus.

Gräther, E 2017. Why working on Chrome made me develop a tool for reading source code. Viitattu 20.5.2018, https://medium.com/@egraether/why-working-on-chrome-made-me-develop-a-tool-for-reading-source-code-7111ba21a6f0.

Guzel, B 2011. Top 15+ Best Practices for Writing Super Readable Code. Viitattu 3.6.2018, https://code.tutsplus.com/tutorials/top-15-best-practices-for-writing-super-readable-code--net-8118.

InfoQ 2017. Dead Code Must Be Removed. Viitattu 12.5.2018, https://www.infoq.com/news/2017/02/dead-code.

Palakollu E, 2016. React and Flux life cycle with JSX, React Router and Jest Unit Testing. Viitattu 27.5.2018, https://www.slideshare.net/EswaraP/react-flux-react-router-and-jest.

ReactJS 2018. Docs. Viitattu 12.5.2018, https://reactjs.org/docs.

Sonmez, 2017. What Software Developers Should Know About Testing and QA. Viitattu 17.6.2018, https://simpleprogrammer.com/software-developers-know-testing-qa/.

Tartakovsky, M, 2016. 12 Foolproof Tips for Finding Focus. Viitattu 8.7.2018, https://psychcentral.com/lib/12-foolproof-tips-for-finding-focus/.

University of Helsinki & Reaktor 2018. Elements of AI. Viitattu 1.7.2018, http://www.elementsofai.com/fi.