# JWT 2018 harjoitustyö

Tekijä: Arttu Ylhävuori

Osoite: http://www.sis.uta.fi/~ay98625/jwt/ht/index.html

Palautuspäivä: 28.5.2018

## 1. Harjoitustyön toteutus

Harjoitustyön toteutukseni noudattaa hyvin pitkälti annettua ohjeistusta ja sen mallikuvia. Käytin harjoitustyöhön aikaa noin 15 tuntia, jotka ajoittuivat viikoille 19–22. Olen testannut harjoitustyöni toimivuuden käyttäen Google Chrome (versio 68.0.3438.3) -verkkoselainta.

Aloitin harjoitustyön tekemisen heti ohjelmoimalla harjoitustyön asettelun, jonka pohjana käytin Bootstrapia. Koska ruudukon tekeminen tuntui haasteelliselta tehtävältä, etsin verkosta avukseni avoimen lähdekoodin projektin ristinolla-pelistä (ks. osio 4). Kyseinen mallipohja käyttää osuvasti ulkoasun runkonaan Bootstrapia, mikä vastasi myös omiin tarpeisiini erinomaisesti. Otinkin tästä projektista mallia tehdessäni ruudukkopohjan harjoitustyöhön. Muilta osin asettelusuunnitelmani perustui harjoitustyön tehtävänannossa annettuun ohjeistukseen ja mallikuviin.

Sivuston ulkoasun koodaaminen lähti minulla siitä, että laitoin Boostrapin `.container-fluid` -määrittelyn sivulle, `row`-määrittelyn ja sen sisälle kaksi `div`:iä, joissa on `.col-6` -luokkamäärittely. Tarkistin mallitekstillä, että sivu jakautuu kahteen yhtä suureen osaan – peli- ja kontrollialueisiin.

Tein ensin asettelun osalta kontrollialueen valmiiksi lisäten nimi- ja pistealueet sekä ohjauspainikkeet. Sitten ohjelmoin ristinolla-pelin mallin mukaan pelialueen ruudukon, joka näytti jo ulkoasultaan siltä, mitä ohjeistuskin vaatii.

Tämän jälkeen aloitin JavaScript-koodin ohjelmoimisen lisäämällä `querySelector`:it JavaScript-tiedostoon ja ohjelmoimalla nimenvaihtotoiminnon, joka olikin ensimmäinen valmiiksi saamani toiminnallisuus. Tämän jälkeen lisäsin skriptit ohjauspainikkeille samalla testaten, että jokainen nappi toimii. Samassa yhteydessä oli toimivaa lisätä myös pelihahmo ruudukkoon ja laittaa pelihahmo liikkumaan ruudukossa. Pelihahmon kuvatiedoston (sekä myöhemmin aarteiden kuvatiedostot) päädyin lisäämään samaisen JavaScript-tiedoston kautta, sillä CSS-ohjelmointikielen käyttö ja linkittäminen JavaScript-tiedostoon on tässä yhteydessä mielestäni turhan hankalaa.

Pelihahmon navigoinnin lisäämisen jälkeen olikin luontevaa lisätä aarteita ruudukkoon. Tämä toimenpide kävi myöskin varsin vaivattomasti – sen kun vain lisäsin kuvatiedostojen polut JavaScript-tiedostoon. Sen sijaan helppoa ei ollut pistelaskurin ohjelmoiminen – aikaa meni siihen yllättävän kauan. Tästä lisää osiossa 2.

Harjoitustyön tekemisen viimeiset vaiheet olivat ulkoasun hienosäätöä, koodin siistimistä, uuden pelaajan ohjaus nuolinäppäimillä -toiminnon lisäämistä, aarre- ja pelaaja-kuvakkeiden vaihtamista itse suunniteltuihin kuvakkeisiin, koodin loppusiivousta, aarteiden pisteyttämistä eri arvoisiksi ja tämän harjoitustyöraportin kirjoittamista. Suunnittelin myös vielä lisääväni harjoitustyöhön loput lisäpisteisiin

oikeuttavat lisätoiminnot, mutta en lopulta malttanut.

## 2. Ongelmakohdat

Pistelaskurin ohjelmoiminen oli ongelmallisinta harjoitustyötä tehdessä. Aarteiden lisääminen ja keruu onnistui kyllä helposti, mutta pisteiden kertymisen toteuttaminen ei ottanut onnistuakseen.

Osasin helposti toteuttaa `tarkistaOsuma()` -funktioon tarkistuksen, joka tarkistaa, onko aarre samassa ruudussa pelaajan ja palauttaa `true` tai `false`. En kuitenkaan saanut koodia toimimaan ihan noin vain – jostain syystä edellä mainitun funktion sisällä oleva `onkoAarreRuudussa()` -funktio tuotti jatkuvasti tulokseksi `true`, vaikkei pelaajan kanssa samassa ruudussa ollutkaan aarretta. Koin, että minulla oli vaikea hahmottaa, miten ruudukon koordinaatit tulisi määritellä.

Lopulta katsoessani pienen tauon jälkeen ohjelmakoodia hoksasin, että funktion käyttö vaatii kaarisulkeita `()`, jotka puuttuivat ehtolauseen määrittävästä ehdosta. Sulut lisäämällä ohjelmakoodi alkoi toimia vaatimusten mukaisesti.

## 3. Reflektio

Arvioin onnistuneeni ihan kiitettävästi harjoitustyön tekemisessä. Harjoitustyön minimivaatimus täyttyikin harjoitustyötä tehdessäni yllättävän nopeasti, eli jo hyvissä ajoin ennen esiversion palautusta.

Olen kuitenkin hieman tyytymätön siihen, miten minimivaatimuksen toteuttamisen jälkeen intoni tehdä lisätoimintoja lopahti tyystin. Lopullisen palautuspäivän lähestyessä en oikein enää jaksanut miettiä sitä, miten esimerkiksi aarteet saisi liikkumaan satunnaisesti ruudukossa. Syynä lienee kevätväsymys ja kesäloman lähestyminen – tai sitten vain kesäsuunnitelmat ovat jo tässä vaiheessa niin vahvasti mielessä. Onneksi kuitenkin palautuspäivänä sain pientä puhtia harjoitustyön viimeistelyyn, ja onnistuinkin vielä asettamaan aarteille eri arvot.

Opin paljon JavaScript-ohjelmointikielellä ohjelmoimisesta. Olin aiemmin kammoksunut kyseistä kieltä mieltäen, että se on turhan vaikea itsenäisesti opeteltavaksi kieleksi. Yllätykseksi tämän kurssin ja etenkin tämän harjoitustyön tekemisen aikana olen saanut huomata, että JavaScript-kieli onkin hyvin samankaltainen kieli kuin Java-ohjelmointikieli eikä siten mitenkään erityisen vaikea itsenäisesti opeteltavaksi. Harjoitustyötä tehdessäni kynnys hienosäätää JavaScript-kielellä tehtyä ohjelmakoodia on madaltunut huomattavasti ja onnistumisen riemua on tullut koettua muutamaankin otteeseen. Uskoisinkin, että tämän kurssin jälkeen syvennyn lisää etenkin JavaScript-kielen erikoisuuksiin.

Jos nyt lähtisin tekemään harjoitustyötä uudelleen alusta, tarttuisin hanakammin työn viimeistelyyn ja panostaisin nykyistä enemmän lisätoimintojen suunnitteluun ja ohjelmoimiseen. Tämä harjoitustyö osoitti minulle jälleen sen, miten työt olisi syytä saada valmiiksi hyvissä ajoin ennen palautuspäivää – eikä lähteä tekemään vapaa-ajan projekteja ennen pakollisen projektin valmistumista.

## 4. Lainatut osuudet

Käytin päälähteenäni Jennifer Blandin GitHubissa jakamaa avoimen lähdekoodin *Simple tic-tac-toe game* -projektia (https://github.com/ratracegrad/fcc-tic-tac-toe). Tämä lähde oli siitä kätevä, että se opasti minua, miten Bootstrapia käyttäen pystyy ohjelmoimaan asiallisen näköisen peliruudukon ja miten tämän

ruudukon pystyy saamaan JavaScriptiä käyttäen toimimaan. Kyseisessä lähteessä oli tosin ongelmana se, että se käytti ohjelman toiminnallisuuksien ohjelmoimiseen osittain JavaScript-pohjaista jQuery-kirjastoa. Kirjastojen käyttö oli harjoitustyössä kiellettyä, joten jouduin hieman soveltamaan lähteen tarjoamaa mallia omiin käyttötarkoituksiini.

# 5. Harjoitustyön mielekkyys

Harjoitustyö oli aiheeltaan mielestäni kiinnostava, konkreettinen ja muutenkin kokonaisuudessaan mielekäs. Toki tehtävänannon arviointikriteereissä olevia lisätoimintoja olisi voinut tarkemmin avata muuallakin kuin pelkästään arviointitaulukossa – joka toki oli ulkoasultaan nykyisellään selkeä.

Omalta osaltani harjoitustyön vaikeusaste oli juuri sopiva. Mietin tosin sitä, että osalle kurssilaisista saattoi muutenkin jo pelkän HTML- ja CSS-ohjelmointikielien opettelu olla tämän kurssin haastavinta antia, joten harjoitustyössä oli ehkä turhankin suuressa roolissa JavaScript-kielellä ohjelmointi pelin toiminnallisuuksien toteuttamiseksi. Itselleni HTML- ja CSS-kielillä ohjelmoiminen oli jo ennestään tuttua, joten harjoitustyö auttoi minua juuri sopivasti harjoittelemaan JavaScript-kielen alkeita ja omaksumaan työelämässä tarvittavia *front-end web development* -ohjelmointitaitoja.

Harjoitustyölle annettu aikataulu oli mielestäni juuri sopivan väljä. Koska muut opintoni loppuivat jo toukokuun ensimmäisellä viikolla, minulla oli riittävän paljon aikaa panostaa tämän kurssin viimeisiin tehtäviin, kuten viimeiseen viikkoharjoitukseen ja tähän harjoitustyöhön. Tosin lopullisen version palautukseen on nähdäkseni jätetty huimasti aikaa, ehkä enemmänkin kuin olisi tarpeen. On mielestäni kuitenkin hyvä, että harjoitustyön lopullisen version palautukseen on annettu aikaa kaksi viikkoa minimiversion eräpäivän jälkeen – ainakin ohjelmointiaikaa on riittävästi.

Mielestäni harjoitustyön esiversion vaatiminen oli hyvin perusteltua, sillä se oli hyvä motivaattori ainakin itselleni tenttiin pääsyn varmistamiseksi. Oli helpottavaa huomata, ettei minimiversioon vaadittu tätä raporttia, sillä näin pystyin keskittymään kirjallisen dokumentoinnin sijasta harjoitustyön kannalta oleelliseen, eli suunnittelu- ja ohjelmointityöhön.

# 6. Mahdolliset pisteisiin oikeuttavat osuudet

| |
|---|
| Bootstrapin käyttö<br>Bootstrapin muutamia luokkia on hyödynnetty harjoitustyön ulkoasun suunnittelussa. Käytössä ovat seuraavat HTML-koodista löytyvät luokat: `.container-fluid` (sisältö täyttää koko näytön leveyden), `.col-6` (sivu jakautuu kahteen sarakkeeseen), `.alert .alert-primary` (nimialueen värjääminen siniseksi), `.alert .alert-success` (pistealueen värjääminen vihreäksi) ja `.btn .btn-primary` (suuntapainikkeiden värjääminen sinisiksi). |
| Visuaalinen ulkoasu<br>Olen panostanut visuaaliseen ulkoasuun kuvakkeilla (olen korvannut mallina tarjotut pelaaja- ja aarrekuvakkeet itse valitsemillani kuvakkeilla; aarrekuvakkeissa on neljää eri värivariaatiota) ja taustavärien käytöllä (vihreä peliruudukko, siniset ruutujen ääriviivat, oranssi verkkosivun tausta, sekä nimialueen sininen pohjaväri, pistealueen vihreä pohjaväri ja suuntapainikkeiden siniset pohjavärit). |

Arttu Ylhävuori (98625)

Responsiivisuus
Harjoitustyö on responsiivinen. Olen käyttänyt Bootstrapia apuna responsiivisuuden toteuttamiseksi. Kun selainikkunaa kaventaa, sekä peli- että kontrollialue kapenevat. Lisäksi peli- ja aarrekuvakkeet kutistuvat erilliseen CSS-tiedostoon tekemäni `max-width` -määrittelyn ansiosta. Peli- ja aarrekuvakkeet alkavat pienentyä, kun ruudut kuvakkeiden ympärillä ovat tarpeeksi pieniä selainikkunaa kaventaessa. Kontrollialueen otsikko, nimi- ja pistealueet sekä ohjauspainikkeet jatkuvat kaikki toiselle riville, kun selainikkuna on tarpeeksi kapea.

Pelaajan nimen vaihto
Pelaajan nimen voi vaihtaa vaatimusten mukaisesti. Nimen vaihto ei onnistu, jos syöte on tyhjä tai alle kaksi merkkiä pitkä. Jos syöte on tyhjä tai alle kaksi merkkiä pitkä tai jos syötteen antaminen peruutetaan, käyttäjälle annetaan ilmoitus siitä, mitä ehtoja nimen vaihtamisessa on.

Liikkumisen rajaaminen
Pelaajan liikkuminen pelialueen ulkopuolelle on estetty JavaScript-koodiin tehtyjen ehtojen mukaisesti. Toisin sanoen siis mitään ei tapahdu, jos yritetään ulos pelialueelta.

Liikekomennot
Pelaajaa voidaan liikuttaa sekä painikkeiden (ylös, alas, vasen, oikea) että näppäimistön nuolinäppäimien avulla.

Pelikolikot
Pelissä olevat aarrearkut ovat erivärisiä ja eriarvoisia: `aarre0` (keltainen pohjaväri) on arvoltaan 100$, `aarre1` (sininen pohjaväri) on arvoltaan 150$, `aarre2` (vaaleanpunainen pohjaväri) on arvoltaan 300$ ja `aarre3` (vihreä pohjaväri) on arvoltaan 500$.

Pelikolikoiden paikat
Perustaso: Pelikolikot pysyvät paikoillaan siihen asti, kunnes pelaaja poimii ne astumalla samaan ruutuun.

Peliruudukon luonti
Perustaso: Peliruudukko on luotu HTML:ssä ja se on aina 3x3-ruudukko.