Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Emilia Lahtinen

Unity UI:lla toteutetun käyttöliittymän skaalauksen ongelmia ja ratkaisuja

Case The Walking Dead: Our World -mobiilipeli

Metropolia Ammattikorkeakoulu

Medianomi

Viestinnän tutkinto-ohjelma

Opinnäytetyö

23.4.2019

Tekijä Otsikko Sivumäärä Aika	Emilia Lahtinen Unity UI:lla toteutetun käyttöliittymän skaalauksen ongelmia ja ratkaisuja – Case The Walking Dead: Our World -mobiilipeli 63 sivua 23.4.2019
Tutkinto	Medianomi
Tutkinto-ohjelma	Viestinnän tutkinto-ohjelma
Suuntautumisvaihtoehto	Digitaalinen viestintä
Ohjaaja	Lehtori Juhana Kokkonen

Opinnäytetyössä käsitellään Unity UI -työkaluilla toteutettujen käyttöliittymien skaalautumisen aiheuttamia ongelmia ja niiden ratkaisuja. Työn tavoitteena oli näitä ongelmia ja ratkaisuja taulukoimalla vähentämään skaalautumisen testauksen tarvetta käyttöliittymäsuunnittelijoiden työskentelyssä.

Työ on toteutettu yhteistyössä Next Games -peliyhtiön kanssa The Walking Dead: Our World -mobiilipelin kehittämisen yhteydessä. Opinnäytetyön tekijä toimii käyttöliittymäsuunnittelijana yrityksessä.

Työn toteuttamiseen on käytetty tekemisen tutkimuksen, toimintatutkimuksen ja autoetnografian metodeja. Työssä on käytetty lähteinä kirjallisuutta sekä asiantuntija haastattelua, ja työn aineistoa on kerätty tekijän muistojen pohjalta sekä tuotettu kahdessa syklissä toteutetulla esiintyvyyden havainnoinnilla.

Työn lopputuloksena syntyi kolme taulukkoa, joita voidaan skaalautumisen ongelmien ilmetessä käyttää ratkaisemaan käsillä olevia ongelmia. Havainnoinnin aikana löydetyt yksittäiset ongelmat ja ratkaisut on yleistetty taulukoihin, jotta taulukkojen tietoja voidaan paremmin soveltaa. Taulukoissa on esitetty ongelmien kuvaukset sekä ratkaisuvaihtoehdot vaihe vaiheelta. Taulukot toimivat rinnakkain, joten taulukoissa viitataan toisiinsa tarvittaessa.

Opinnäytetyön aiheesta on yleisesti saatavilla vain vähän kirjallisuutta ja muita lähteitä. Työ tekee näkyväksi hiljaista tietoa, jota on vain Unity UI:lla käyttöliittymien skaalausta toteuttavilla käyttöliittymäsuunnittelijoilla. Työn merkitys on rajattu tietyn ohjelman ja tietyn käyttäjäryhmän tarpeisiin, mutta se avaa aloitteleville tekijöille ja aiheeseen perehtymättömille mahdollisuuden ymmärtää ja soveltaa tätä hiljaista tietoa.

Avainsanat	Unity, UI, käyttöliittymä, suunnittelu, skaalaus, mobiilipelit

Author	Emilia Lahtinen
Title	User Interface Scaling Problems and Solutions with Unity UI – Case The Walking Dead: Our World Mobile Game
Number of Pages	63 pages
Date	23 April 2019
Degree	Bachelor of Culture and Arts
Degree Programme	Degree Programme in Media
Specialisation option	Digital Media
Instructor	Juhana Kokkonen, Senior Lecturer

My thesis deals with problems caused by scaling the user interfaces implemented with Unity UI tools provided by Unity game engine. My goal was to try to reduce the need to test the scaling by creating tables for discovered scaling problems and solutions.

I carried out the research in cooperation with Next Games game company. I work as a user interface designer in the company and I have been involved in the development of The Walking Dead: Our World mobile game.

For the research and the analysis, I used practices from action research and autoethnography methods. I collected the research data from literature and other pre-existing sources, as well as from my work notes and diaries. I also produced data by recording two-cycled self-observational occurrences during the winter 2018 and the spring 2019. My work was also validated by an interview which I had with a more experienced colleague.

As a result of this thesis, I created three tables which can be used to solve the scaling problems. The individual problems and solutions I found during the interval self-observation have been generalized in the tables. The tables contain descriptions of the problems and the possible steps to solve the problems. The tables work together so they refer to each other when necessary.

I chose this topic because in general, there is very little literature available on the subject of this thesis. My work brings out tacit information which is mostly available for user interface designers who implement interfaces and scaling with Unity. The importance and benefits of my thesis is limited to the needs of a specific user group which uses a specific program. But it opens up the ability for novice and inexperienced people to understand and apply this tacit knowledge in their work.

Keywords	Unity, UI, user interface, design, scaling, mobile games

Sisällys

1 Johdanto

Työskentelen käyttöliittymäsuunnittelijana *Next Games* -peliyhtiössä, ja työhöni kuuluu suunnittelun lisäksi myös käyttöliittymien toteutus Unity-pelimoottorilla. Unity on kehittänyt käyttöliittymien toteutusta varten erilaisia työkaluja, joita kutsutaan yhteiseltä nimitykseltään Unity UI:ksi. Aloittaessani Next Games -peliyhtiössä, liityin mukaan tekemään *The Walking Dead: Our World* -mobiilipeliä.

Koska olin tekemässä mobiilipeliä, käyttöliittymien skaalautumisen haasteet eri resoluutioisten mobiililaitteiden välillä tulivat myös tutuksi. Tässä opinnäytetyössä käsittelen Unity UI:lla toteutettujen käyttöliittymien skaalauksen ongelmia ja avaan niihin löytämiäni ratkaisuja.

Havaitsin, että välillä suunnittelijoilla ja koodareilla voi olla vaikeuksia toteuttaa Unitylla responsiivinen käyttöliittymä, joka skaalaisi saumattomasti markkinoilla olevien mobiililaitteiden resoluutioden välillä. Unityn virallinen manuaali tarjoaa käyttöliittymän skaalauksesta tietoa varsin niukasti, vaikka Unityn (2018) mukaan Unity-pelimoottoria käyttää 47 prosenttia mobiilipelien kehittäjistä.

Suunnittelijan on vaikeaa tietää ilman testausta, miltä Unitylla toteutettu käyttöliittymä näyttää, kun se esitetään resoluutioltaan pienemmällä tai suuremmalla laitteella. Skaalauksen testaaminen vie käyttöliittymäsuunnittelijoiden aikaa pois muulta kehitystyöltä, kuten visuaaliselta suunnittelulta ja käytettävyyden parantamiselta. Tarkoitukseni oli opinnäytetyölläni pyrkiä vähentämään testauksen tarvetta taulukoimalla skaalautumisen ongelmia sekä niihin sopivia ratkaisuja. Suunnittelija voisi sitten etsiä ongelman kohdatessaan taulukoista mahdollisen ratkaisun.

Koska teen töitä tietokoneen ääressä, pyrin vapaa-ajallani välttämään ruudun ääressä oleskelua, joten harrastan erilaisia käsitöitä vaatteiden ompelusta resiinin valamiseen. Käsillä tekeminen heijastuu suoraan arvomaailmaani, ja siksi työelämässäkin olen enemmän käsityöläinen. Arvostan manuaalista tekemistä enemmän kuin automatiikkaa ja toimivuus on minulle tärkeää viimeistellyn ulkoasun aikaan saamisessa. Saatan tarkastella käyttöliittymää samaan tapaan kuin saviruukkua. Sen tulee olla visuaalisesti laadukas, mutta sen pitää olla myös toimiva eri käyttötarkoituksissa, päätti käyttäjä sitten istuttaa kukkia tai säilyttää esineitä. Saviesineiden hajoaminen raakapoltossa on itselleni

saman tason ongelma kuin se, että käyttöliittymän ulkoasu muuttuu skaalatessa liikaa verrattuna natiiviresoluutioon.

Opinnäytteeni on arvojeni mukaisesti tehty kädet savessa, eli lähestyn aihettani vahvasti tekemisen kautta ja käytän apunani tekemällä tutkimisen, toimintatutkimuksen ja autoetnografian metodeja, jotta saan tekemällä opitun nostettua myös tutkimuksellisesti pätevällä tavalla esille.

Opinnäytteeni ei ole niinkään tekninen työ, vaikka aiheeni onkin tiukasti rajattu yhden ohjelman sisällä olevien työkalujen ongelmiin ja näiden ongelmien ratkaisemiseen. Opinnäytetyöni painopiste on poikkeuksellisesti suunnittelijan tekemässä käsityössä, jolla voidaan toteuttaa laadukkaasti skaalautuva käyttöliittymä. Tätä laadun abstraktia käsitettä ymmärtävät lähinnä toiset suunnittelijat, ja tämän vuoksi aiheestani on vaikeaa kirjoittaa niin, että se avautuisi aiheeseen perehtymättömällekin lukijalle. Olen siksi pyrkinyt selittämään asiat maanläheisesti ja käyttämään mahdollisimman paljon esimerkkikuvia sekä tekstin värikoodausta, jotta teksti ja kuva yhdistyisivät lukijan mielessä.

Opinnäytetyöni toisessa luvussa kerron menetelmäni työn tekemiseen sekä selitän keskeiset käsitteet. Kolmannessa luvussa käyn läpi työni lähteet sekä aineiston, joka koostuu The Walking Dead: Our World -projektin aikana tekemistäni muistiinpanoista, autoetnografisten välineiden avulla kerätystä havainnointiaineistosta sekä pelin käyttöliittymistä. Neljännessä luvussa esittelen lyhyesti Unity UI:n keskeisimpiä skaalauksen työkaluja, jotta työtä on helpompi seurata Unitya tuntematta. Viidennessä luvussa analysoin skaalauksen ongelmat sekä ongelmien ratkaisuvaihtoehdot. Kuudennessa luvussa esittelen johtopäätöksenä löydetyt ongelmatyypit ja niiden ratkaisut taulukoituina, ja lopuksi on vielä lyhyt yhteenveto.

2 Menetelmät ja käsitteet

Tässä luvussa määrittelen tarkemmin opinnäytetyöni tekemiseen käytetyt menetelmät. Ensin käsittelen työn tavoitteet ja rajaukset. Seuraavaksi esittelen aineiston keräyksen ja analyysin menetelmät. Menetelmät jakautuvat ala-luvuissa tekemällä tutkimiseen, toimintatutkimukseen sekä autoetnografiaan. Sen jälkeen selitän hermeneuttisen kehän merkitystä työn toteuttamisessa ja viimeiseksi avaan työn aiheeseen liittyvät käsitteet.

2.1 Tavoitteet

Työ keskittyy Unityn valmiiksi tarjoamiin käyttöliittymätyökaluihin eli yhteiseltä nimitykseltään Unity UI:hin. Rajaan tutkimuksesta pois Unitylle saatavat lisäosat sekä ohjelmoinnilla mukautetut työkalut. Lisäksi, koska työ keskittyy nimenomaan skaalaukseen, työn pääpaino on Unity UI:n skaalauksen työkaluissa. En myöskään tutkimuksessa ota kantaa käyttöliittymän visuaaliseen suunnitteluun.

Rajaan työn keskittymään yksinomaan mobiililaitteille tehtäviin pelien käyttöliittymiin, joten työstä jäävät ulos muut pelialustat ja käyttöliittymät. Lisäksi aihetta tarkastellaan nimenomaan käyttöliittymäsuunnittelun näkökulmasta, joten en ota kantaa metodien vaikutuksiin ohjelmoinnissa tai optimoinnissa.

Työ ei tarjoa valmiita vastauksia skaalauksen toteuttamiseen. Työn tarkoitus on antaa suunnittelijoille selostus siitä, mitä skaalauksen ongelmia voi tulla vastaan, ja tarjota mahdollisia ratkaisuja, joista suunnittelija voi tilanteestaan riippuen valita sopivan.

2.2 Tekemällä tutkiminen

Kuten johdannossakin kerroin, olen käsityöihminen, joten käytännönläheisestä tekijän roolista siirtyminen enemmän tutkivaan ajatteluun ei ole käynyt käden käänteessä tätä opinnäytetyötä tehdessä. Pirkko Anttilan kirjoittama *Tutkiva toiminta ja ilmaisu, teos, tekeminen* (2006) on ollut avainasemassa tässä muutoksessa. Anttila (2006, 423) kirjoittaa, että vallalla on hämmennys siitä, voidaanko käytännön työtä pitää jonkinlaisena tutkimusmenetelmänä sinänsä. Koska on vaikeaa tajuta, kuinka työtä voidaan soveltaa tutkimuksellisesti, se voi johtaa tavanomaisten akateemisten tutkimusmenetelmien liialliseen käyttöön käytännön työn kustannuksella. Tämä taas voi johtaa käytännön työn painoarvon väheksymiseen. (Anttila 2006, 424.)

Anttila (2006, 424) kertoo kirjassaan Schönin reflektiivisen toiminnan teoriasta, johon kuuluu toteamus, että aikaisemmat kokemukset aikaansaavat esimerkkejä, mielikuvia, ymmärrystä ja toimintaa pikemminkin kuin yleistettyjä teorioita, menetelmiä, tekniikoita tai työvälineitä. Reflektiivisen toiminnan prosessin uskotaan olevan subjektiivinen. Siihen kuuluu näkemys, jonka mukaan luova tekijä hakee vahvistusta paremmin kuin tunnustusta toiminnan soveltuvuudelle. Reflektio on keskeistä ammatilliselle kehittymiselle. Anttilan mukaan Schön näkee reflektion primaarina kognitiivisena mekanismina, jonka

avulla voidaan käsitellä odottamattomia mekanismeja. Jotta sekä tutkimusmielessä että oppimismielessä reflektion seuraamuksista saataisiin tarkoitettu hyöty irti, reflektion sekä odotetut että odottamattomat tulokset pitäisi tallentaa systemaattisesti. (Anttila 2006, 425.)

Kiinnostuttuani tekemällä tutkimisesta valitsin opinnäytetyöni tiedon tallentamis- ja analysointiprosessin menetelmäksi Anttilan esittelemän aktiivisen dokumentaation. Aktiivinen dokumentaatio paljastaa teoreettiset, henkilökohtaiset ja käytännölliset tarkoitusperät sekä niihin liittyvät ja esille kohonneet vaikeudet ja ongelmat (Anttila 2006, 426). Anttilan mukaan aktiivista dokumentointia voidaan pitää erityisenä käytännöntavoitteiselle tutkivalle toiminnalle soveltuvana menetelmänä. Anttila (2006, 426) lainaa teoksessaan muotoilun tutkijan, Nancy De Freitasin ajatusta siitä, että aktiivisen dokumentoinnin avulla voidaan seurata työn edistymistä, erottaa selvästi hyvin edistyviä sekä ongelmallisia vaiheita sekä pelkistää aineistoa yleiselle tasolle saamista varten.

Aktiivisen dokumentoinnin merkitys osana tekemällä tutkimista korostuu opinnäytetyössäni siksi, että se on keino saada kuva tehtyjen ratkaisujen pätevyydestä (Anttila 2006, 221). Anttila (2006, 221) toteaa lisäksi, että vaikka aktiivinen dokumentointi ei ole varsinainen tutkimusmenetelmä, se on menetelmä nähdä ideoiden kehityskaari. Sen avulla sain poimittua arkisesta aherruksesta ongelmanratkaisujen kehityksen. Tällä tavalla havainnoinnin aikana syntyneestä aktiivisesta dokumentaatiosta pystyi erottamaan selvästi mistä tilanteesta ongelmaa oli lähdetty ratkaisemaan, mitä vaiheita ongelman ratkaisemiseksi piti käydä läpi ja mihin ratkaisuun päädyttiin, jos päädyttiin. Työtä tehdessä tämä on aivan arkinen prosessi, mutta jos sen äärelle ei pysähdy ja sitä ei aktiivisesti dokumentoi, saman prosessin voi joutua käymään läpi turhaan uudelleen ja uudelleen.

2.3 Toimintatutkimus

Anttilan teoksen kautta tutustuin myös toimintatutkimukseen. Toimintatutkimus on ns. pehmeisiin menetelmiin lukeutuva tutkimusmenetelmä, jonka painopisteenä ei ole niinkään saada yleistettävää tietoa kuin täsmällistä tietoa tiettyä tilannetta ja tarkoitusta varten. Toimintatutkimuksen tarkoituksena on kehittää uusia taitoja tai uutta lähestymistapaa johonkin tiettyyn asiaan sekä ratkaista ongelmia, joilla on suora yhteys johonkin käytännölliseen toimintaan. Se sopii hyvin tilanteisiin, joissa toiminnan avulla pyritään muuttamaan jotakin ja samanaikaisesti lisäämään sekä ymmärrystä että tietoa muutosta kohtaan. Tutkittavat ovat aktiivisia osallistujia prosessissa, tutkimus suuntautuu käytäntöön

ja se on ongelmakeskeistä. Toimintatutkimus on suoraan yhteydessä työ- tai toimintatilanteeseen, ja se tarjoaa järjestelmällisen kehyksen ongelmaratkaisutilanteeseen. Se on laskettavissa empiirisiin menetelmiin, koska siihen liittyy havainnointia. Verrattuna autoetnografiaan toimintatutkimus ei käytä hyväkseen aikaisempien kokemusten muistelua vaan liittyy “tässä ja nyt” -kokemuksiin. Toimintatutkimuksella ei tavoitella yleistettäviä tuloksia, vaan sen tavoitteet liittyvät tiettyyn tilanteeseen ja tuloksilla on merkitystä vain asianomaisille. Tutkijan rooli on verrattavissa konsultin rooliin: hän auttaa “toimijoita” tiedostamaan ja ratkaisemaan kehittämisen kohteen ongelmia ja selviytymään ratkaisemattomien ongelmien parissa. (Anttila 2006, 439–447.)

Toimintatutkimuksen kirjaimellinen noudattaminen ei ollut opinnäytetyöni tarkoitus, mutta työni on kulkenut hyvin paljon toimintatutkimuksen periaatteiden ja vaiheiden mukaan. Ensin määrittelin opinnäytetyöni ongelmat ja tavoitteet. Sen jälkeen etsin aiheesta kirjallisuutta ja muuta lähdemateriaalia ja sen pohjalta pyrin löytämään jo olemassa olevaa valmista tietoa aiheesta. Sen jälkeen määrittelin havainnoinnilleni raamit ja lopuksi analysoin kootun tiedon tämän työn lopputulokseksi. Anttilan toimintatutkimuksen vaiheet käyvät läpi samoja vaiheita. Ensin määritellään ongelma tai asetetaan tavoitteet. Sitten tehdään katsaus kirjallisuuteen tai muuhun lähdemateriaaliin, jotta voidaan todeta onko aiemmin kohdattu saman tyyppisiä ongelmia. Tämän jälkeen määritellään testattavat ongelmat tai lähestymistavat sekä suunnitellaan toimintatutkimuksen asetelma. Sitten määritellään arviointikriteerit ja muut palautteen saamisen muodot ja lopuksi analysoidaan koottu tietoaines ja evaluoidaan tulokset. (Anttila 2006, 443.)

Toimintatutkimuksen validiteetin edellytyksiä ovat muun muassa korkeatasoinen osaaminen ja havainnointikyky, ryhmässä työskentely, aineiston kerääminen sykleissä sekä syklien suunnittelu ja monen erilaisen tiedon vuorovaikutus (Anttila 2006, 446). Vaikka opinnäytetyöni ei ole puhtaasti toimintatutkimus, koen työn täyttävän monia näistä kriteereistä. Ensinnäkin havainnointikykyä voi toimintatutkimuksen aikana kehittää, ja olen saavuttanut tekemiseni aikana paljon korkeamman havainnoinnin tason niin itsessäni kuin muissakin. Vaikka olen tehnyt opinnäytetyöni pääasiassa yksin, olen havainnoinnin aikana sisällyttänyt työtoverini osallisuuden aineistoon. Näin pystyn osoittamaan heidän antamansa panoksen aineiston keräyksessä. Lisäksi olen validoinut työni sisällön mentorillani. Havainnointiaineiston kerääminen tapahtui kahdessa syklissä, jotta pystyin ensimmäisen syklin jälkeen tarkastelemaan jo keräämäni aineiston sisällön ja suunnittelemaan seuraavan syklin tavoitteita sen mukaisesti. Näin havainnointiaineistoni puutteelli-

set kohdat tuli huomattua ajoissa. Olen työtäni varten tutustunut aiheeseen liittyvään kirjallisuuteen sekä muihin lähteisiin, sekä kerännyt aineistoa muistojen pohjalta, havainnoimalla sekä haastattelulla.

2.4 Autoetnografian välineet

Pelialalla toimivana suunnittelijana, joka työskentelee skaalautumisen kanssa noin viisi päivää viikossa, olen kokemuksen kautta saavuttanut tietoa, jota on lähinnä kollegoillani eikä heistäkään kaikilla. Koska tietoni käyttöliittymän skaalautumisesta perustuvat pitkälti mentorini opetuksiin ja kokemuksen kautta saatuun tietoon, koen, että autoetnografiset menetelmät ovat tarpeen tämän tiedon saattamiseksi näkyväksi.

Autoethnography as a Method -kirjan kirjoittaja Heewon Chang (2008, 46–51) kirjoittaa, että autoetnografien on oltava valmiita kaivamaan syvemmin muistojaan löytääkseen niistä rikkaat yksityiskohdat ja tuomaan ne esille lajiteltavaksi ja yhdisteltäväksi sekä kontekstualisoimaan ne sosiokulttuurisessa ympäristössä. Autoetnografia tarjoaa tutkijoille ja lukijoille helposti lähestyttävän tutkimusmenetelmän, jossa yhdistyy lukijoita houkutteleva henkilökohtaisesti kiinnostava kirjoitustyyli sekä tutkijoille helppo pääsy ensisijaiselle tietolähteelle, koska lähde on tutkija itse. Alkuperän tuntemus antaa autoetnografialle etua myös tiedonkeruuseen, analysointiin sekä tulkintaan. (Chang 2008, 51–54.)

Vaikka hyödynnän autoetnografisia menetelmiä, työni ei kuitenkaan ole autoetnografinen sen perinteisessä merkityksessä. Changin (2008, 46) mukaan autoetnografisten tarinoiden odotetaan heijastelevan, analysoivan ja tulkitsevan laajempaa sosiokulttuurista kontekstia ja autoetnografisen itsetarkastelun tavoitteena on usein kulttuurinen ymmärrys. Oma työni taas on opas Unity UI:lla skaalaamisesta eikä liity kulttuuriin varsinaisesti lainkaan. Mutta jotta voin käsitellä kokemaani, käytän autoetnografisia menetelmiä löyhästi hyödykseni. Sisällytän työhöni lisäksi virallisia lähteitä ja kirjallisuutta, mutta aiheeni tarkkarajaisuuden vuoksi ulkoapäin saatavaa tietoa on hyvin rajallisesti. Tämänkin haasteen vuoksi autoetnografiset välineet ovat tarpeen tässä opinnäytetyössä.

2.4.1 Esiintyvyyden tallennus havainnoinnilla

Tavoitteeni esiintyvyyden tallennuksen havainnoinnilla oli kirjata ylös työpäivien aikana esiintyneitä skaalautumisen ongelmia sekä mahdollisia ratkaisuja näihin ongelmiin. Va-

litsin tämän menetelmän kerätäkseni käyttäytymiseni, ajatukseni, tunteeni ja vuorovaikutukseni muiden kanssa sillä hetkellä, kun tilanteet tapahtuivat (Chang 2008, 89). Havainnointi tapahtui kahdessa syklissä, joista ensimmäinen sijoittui 19.11.–14.12.2018 ja toinen 7.1.–1.2.2019 välille. Pyrin näiden syklien aikana selvittämään, millaisia ongelmia nousi esille ja millaisia ratkaisuja niihin löytyi vai jäivätkö ongelmat ratkaisematta. Kirjasin ylös myös, kenen kanssa olin tekemisissä sekä missä ja mihin aikaan skaalautuminen nousi esille. Havainnoinnilla aineistooni syntyi nykyhetkessä kerättyä, tuoretta dataa, joka ei ole vain muistin varassa olevaa aineistoa.

2.4.2 Autoetnografinen haastattelu

Chang kuvaa mentorit viisaiksi ja luotettaviksi oppaiksi ja neuvojiksi, joilta on opittu uusia tietoja, taitoja, periaatteita, viisautta tai näkökulmia. Kulttuurisen tiedon hankinta ja siirto tapahtuu usein mentorien ja mentoroitavien välillä, koska mentorit kutsuvat tahallaan tai tahattomasti oppilaita jakamaan tietoa, taitoja ja näkökulmaa kulttuuriryhmistään. (Chang 2008, 79.) Kun aloitin työskentelemään Next Games -peliyhtiössä, sain mentorikseni Jani Reinikaisen, joka opetti minulle Unityn käytön samalla, kun kehitimme The Walking Dead: Our World -mobiilipeliä. Meillä on kehitystyöstä yhteisiä muistoja, mutta perspektiivimme olivat hyvin erilaiset. Olin vasta taloon tullut ja vailla kokemusta, ja hän taas on monitaituri, jolla on vuosien kokemus takanaan. Mentorini teki ratkaisevat päätökset The Walking Dead: Our World -pelissä käytetyistä käyttöliittymän skaalauksen menetelmistä, joten koin välttämättömäksi haastatella häntä mentorina opinnäytetyöhöni liittyen ja saada häneltä palautetta ja uutta perspektiiviä työhöni.

Autoetnografisissa haastatteluissa tarkoituksena on kerätä kontekstuaalista informaatiota, joka vahvistaa jo olemassa olevaa dataa, täyttää mahdollisia aukkoja haastattelijan tiedoissa ja auttaa hylkäämään itsetutkiskelun kautta luotuja tietoja, jotka eivät laajemmassa kontekstissa pidä paikkaansa (Chang 2008, 104). Reinikainen toimi haastattelun kautta työlleni asiantuntijalähteenä, joten sain validoitua työssäni olevat tiedot kokeneemmalla henkilöllä. Haastattelu toteutettiin kasvokkain yksilöhaastatteluna ja strukturoimattomana. Syy tähän on se, että koin saavani strukturoimattomalla haastattelulla enemmän irti keskustelusta, koska Changin (2008, 105) mukaan strukturoimattoman haastattelun etuna on se, että molemmat osapuolet ovat hallitsevassa asemassa haastattelussa, joten keskustelu pysyy vapaamuotoisena ja avoimena. Haastateltava on aiheesta enemmän perillä kuin minä, joten koin, että strukturoitu haastattelu ei tuottaisi

haluttuja tietoja. Haastattelussa kysyin ensin vapaamuotoisia, keskustelevia kysymyksiä ja sen jälkeen haastateltavan vastausten perusteella tarkempia kysymyksiä.

2.5 Hermeneuttinen kehä

Kirjassaan *Tutkiva toiminta* Pirkko Anttila (2006, 279) selittää, että tutkijan on käytävä läpi aineistoaan monta kertaa koettaessaan vapautua omista esteistään ymmärtää tutkimuskohdetta. Tätä aineiston ympärillä kiertämistä voidaan kutsua myös hermeneuttiseksi kehäksi. Hermeneuttinen kehä lähtee aina tietyistä lähtökohdista ja palaa takaisin niiden oivaltamiseen ja ymmärtämiseen (Anttila 2006, 279). Hermeneuttinen kehä ja "vaeltelu edestakaisin" datan ja teorian välillä, kuten Anttila (2006, 279) asiaa kuvaa, ovat omassa työssäni tulleet tutuksi sillä, että vasta työn edetessä ongelmien ja ratkaisujen luonne alkoi hiljalleen avautua, mutta samalla löytyi paljon uusia solmuja. Analyysivaiheessa jäsentelin aineistoa monin erilaisin tavoin ja useampina päivinä, jotta aika ehtisi tehdä tehtävänsä ajattelulle. Tämän vaeltelun seurauksena löysin analyysissa esiteltävän tavan teemoitella ongelmat ja ratkaisut.

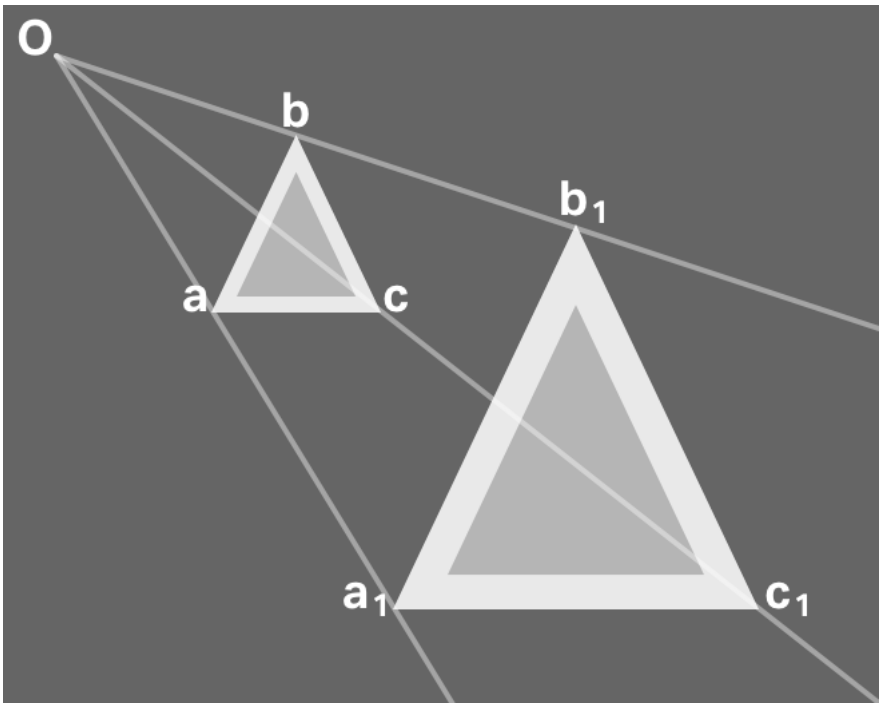
2.6 Käsitteet

Koska kyseessä on opinnäytetyö aiheesta, joka edellyttää lukijalta tiettyä tietopohjaa käyttöliittymäsuunnittelusta, pelien kehityksestä sekä Unity-pelimoottorista, esittelen tässä alaluvussa muutamia oleellisia käsitteitä aiheelle.

Käyttöliittymä määritellään Kielitoimiston sanakirjassa (2018) välineiksi ja toiminnoiksi, joilla käyttäjä viestii tietojärjestelmän kanssa. Käyttöliittymäsuunnittelu on siis tästä määrittelystä johdettuna toimintaa, jolla suunnitellaan ja toteutetaan välineet ja toiminnot käyttäjälle tietojärjestelmän kanssa viestimiseen. Käyttöliittymästä käytetään lyhennettä UI, joka tulee englannin kielen sanoista *user interface*.

Responsiivisuus tarkoittaa Kielitoimiston sanakirjan (2018) mukaan sitä, että päätelaitteen sisältö mukautuu käyttäjän käyttötapaan. Esimerkiksi kun verkkosivun leveyttä kavennetaan tai suurennetaan selaimessa, verkkosivun sisältö mukautuu selaimen uuteen kokoon. Tämän lisäksi responsiivisuuden voi nähdä merkitsevän tapaa tehdä visuaalista sisältöä; responsiivinen suunnittelu on yhtä paljon responsiivinen sisältö itsessään kuin tapa tehdä responsiivinen sisältö. (StackExchange 2015.)

Skaalautumisen selittäminen on haasteellista, sillä kyseessä on yleinen termi, jolla on useita määritelmiä. Käytän tässä yhteydessä apuna geometriassa esiintyvää homotetiaa selittämään skaalautumista. Lehtori Jani Hannulan (2015) mukaan homotetia on yhdenmuotoisuuskuvaus, joka kuvaa kuviot yhdenmuotoisiksi kuvioiksi. Selittääkseni tätä paremmin olen kuviossa 1. kuvannut kaksi kolmiota, jotka ovat erikokoisia. Pienempi kolmio on venytetty suuremmaksi käyttämällä apuna kolmea viivaa, jotka lähtevät pisteestä O. Tätä pistettä kutsutaan homotetiakeskukseksi (Hannula, 2015). Homotetiakeskuksen ja apuviivojen avulla kolmioita voidaan skaalata eri kokoisiksi siten, että kolmiot pysyvät yhdenmuotoisina.



Kuvio 1. Homotetian ja homotetiakeskuksen havainnekuva.

Homotetiaan pohjaten käyttöliittymien skaalautumisen tarkoituksena on saada käyttöliittymät pysymään yhdenmuotoisina, vaikka käyttöliittymiä tarkasteltaisiin erikokoisilla näytöillä. Eli optimaalista olisi löytää niille yhteinen homotetiakeskus. Kuitenkin käyttöliittymän skaalautumiseen liittyy muutakin kuin yhdenmuotoisuus. Käyttöliittymän skaalautumisella on myös laadullisia ominaisuuksia, kuten esimerkiksi käytettävyys ja luettavuus, jotka tulee ottaa huomioon skaalautumista suunnitellessa.

Esteetöntä opiskelua edistävä ESOK-verkosto määrittelee sanastossaan käytettävyyden muun muassa tarkoittamaan menetelmä- ja teoriakenttää, jonka kautta käyttäjän ja

laitteen yhteistoimintaa pyritään saamaan tehokkaammaksi ja käyttäjän kannalta miellyttävämmäksi. Käytän tässä työssä tätä määritelmää käytettävyydestä, koska käyttöliittymäsuunnittelijan näkökulma korostaa pyrkimystä parantaa käytettävyyttä.

Luettavuus määritellään Kielitoimiston sanakirjassa (2018) tekstin selkeytenä, ymmärrettävyytenä tai helppolukuisuutena. Skaalauksesta puhuttaessa helppolukuisuus on oleellinen määritelmä. Huonosti toteutettu skaalaus voi tehdä tekstin koosta niin pientä, että lukemisesta tulee mahdotonta.

Resoluutio on suure, joka ilmaisee tässä tapauksessa kuvan esitystarkkuuden (Kielitoimisto 2018). Mittayksikkönä käytetään yleensä pistettä tai pikseliä, jonka määrä pituusyksiköllä kertoo, miten tarkka kuva on (AfterDawn 2018). Esimerkiksi kuva, jonka resoluutio on 1000 pikseliä per tuuma on tarkempi laadultaan kuin kuva, jonka resoluutio on 100 pikseliä per tuuma.

Natiiviresoluutio taas on näytön optimaalisin resoluutio eli kuva on pikselintarkka (AfterDawn 2018). Natiiviresoluutiossa suunnitellessaan käyttöliittymäsuunnittelija voi olla varma siitä, että käyttöliittymä näyttää täsmälleen samalta lopullisella päätelaitteella. Natiiviresoluutiossa kuvaa kun ei venytetä lainkaan pysty tai vaakasuunnassa (AfterDawn 2018).

Kuvasuhteella tarkoitetaan kuvan leveyden ja korkeuden suhdetta. Esimerkiksi 1:1 kuvasuhteen omaava kuva on neliö ja 1:2 kuvasuhteen kuva on kaksi kertaa leveämpi kuin korkea. (AfterDawn 2018) Kuvasuhteet vaihtelevat mobiililaitteissa laajasti valmistajasta riippuen, joten siksi tämä käsite on oleellinen käyttöliittymän skaalauksessa.

Elementti voi kielitoimiston (2018) mukaan olla esimerkiksi perusosa tai valmisosa. Kun puhun tässä työssä elementeistä, tarkoitan Unity UI:hin kuuluvia valmisosia, joilla tehdään käyttöliittymiä. Tällaisia valmisosia voivat olla käyttäjille näkyvät kuvat, napit ja tekstit. Elementtejä ovat myös ne osat, jotka eivät näy käyttäjälle, mutta jotka komponenttien avulla hallitsevat käyttöliittymän hierarkiaa.

Komponentti tarkoittaa kokonaisuuden osaa, osatekijää tai rakenneosaa (Kielitoimisto 2018). Tässä työssä komponentti tarkoittaa elementin sisällä olevaa osaa, joka ohjaa elementin toimintaa määrittelyillä eli asetuksilla. Komponenteilla voidaan esimerkiksi hallita sitä, minkä värinen tai kokoinen elementti on tai miten elementti hallitsee sen sisällä

olevia muita elementtejä. Ero elementin ja komponentin välillä on se, että elementti on käyttöliittymän osa ja komponentti on elementin osa.

3 Aineisto ja lähteet

Tässä luvussa esittelen opinnäytetyöni tekemiseen käytetyt lähteet ja aineistot. Aloitan lyhyellä kirjallisuuskatsauksella, ja sen jälkeen esittelen menetelmät, joilla keräsin aineistoa menneisyydestä ja nykyisyydestä. Lisäksi käyn läpi haastattelun käsittelyn lyhyesti.

3.1 Kirjallisuus

Etsiessäni kirjallisuutta Unity-manuaalin tueksi löysin vain kolme teosta, jotka keskittyvät pääasiassa Unity UI:hin. Esittelen teokset lyhyesti ja vertailen niitä hieman keskenään, koska vähäisen kirjallisuuden vuoksi mielestäni on syytä eritellä, mitä kukin kirja on lähteistöön antanut.

Francesco Sapion kirjoittama *Unity UI Cookbook* sisältää reseptejä pelien käyttöliittymän eri osien toteuttamiseen ja käsittelee siksi Unity UI:n työkaluja ja hieman skaalautumista. Aiheeseeni liittyvää tietoa kirjassa on niukasti, koska kirja painottuu C#-ohjelmointikielen käyttöön Unity UI:n työkalujen rinnalla. Siksi viittaan tähän teokseen vain vähän opinnäytetyössäni. Sapiolla on kuitenkin muutamia, mahdollisesti juuri ohjelmoinnin kautta saatuja oivalluksia, jotka laajensivat omaa ymmärrystäni komponenttien toiminnasta, ja haluan siksi viitata teokseen työssäni.

Ashley Godboldin teos *Mastering UI Development with Unity* käy perusteellisesti läpi Unity UI:n työkalut ja tarjoaa vaihe vaiheelta eteneviä reseptejä käyttöliittymien eri osien toteuttamiseen. Vaikka tässäkin teoksessa suositellaan lukijan ymmärtävän C#-ohjelmointikieltä, Godbold käyttää mielestäni ohjelmointia enemmän lisämausteena, kun taas Unity UI on teoksen täyttävä pääruoka. Tämä mielipide perustuu siihen, että en ole koskaan ohjelmoinut C#-kielellä, mutta toisin kuin Sapion teosta lukiessani, Godboldin kirjassa en kokenut tätä ongelmaksi ja huomasinkin vaatimuksen kielen ymmärtämisestä vasta kun olin jo lukenut kirjan.

Simon Jacksonin kirjoittama *Unity 3D UI Essentials* on hyvä lisäys lähteisiin, koska kirjan alussa Jackson palaa ajassa taaksepäin ja kertoo Unityn työkalujen historian ennen kuin

esittelee uudet ominaisuudet. Tämän kirjan ansiosta myös minä, joka olen hypännyt alalle vasta 2018, pystyn ymmärtämään nykyisten Unity UI:n työkalujen merkitystä verrattuna niiden edeltäjiin. Historian lisäksi Jackson käy läpi Unity UI:n työkalut ja painottaa hieman Godboldia enemmän hierarkian rakentamista, ankkurointia ja skaalautumista. Luin Jacksonin teoksen Apple Books -applikaation kautta ePub-versiona eli sähkökirjana. Apple Books -applikaatiossa teoksien sivunumerot vaihtelevat kirjasinkoon mukaan, joten sen vuoksi Jacksoniin viitatessani kerron sen luvun ja kappaleen otsikon, josta lainaus on.

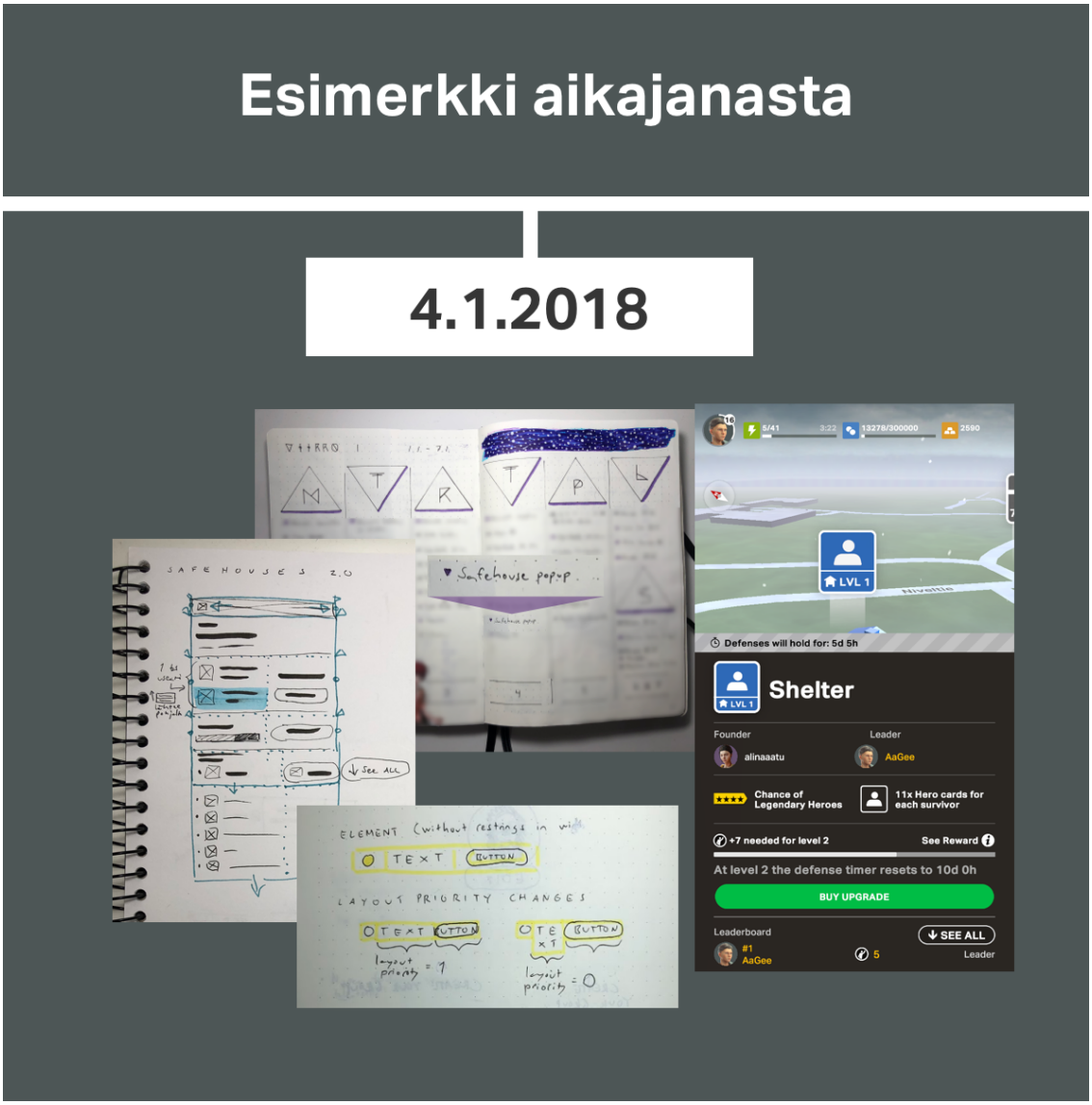
3.2 Muistot

Chang (2008, 71) toteaa, että autoetnografit avoimesti tiedostavat omat henkilökohtaiset muistonsa pääasiallisena aineistona tutkimuksissaan. Muisti kuitenkin valikoi, muokkaa, rajoittaa ja vääristelee menneisyyttä. Se usein paljastaa vain osatotuuksia ja on joskus epäluotettava ja ennalta arvaamaton. Siinä missä jotkin kaukaiset muistot säilyvät kirkkaina, lähimenneisyyteen kuuluvat muistot saattavat haalistua nopeasti ja hämärtää todellista aikaväliä tapahtumien välillä. Yksityiskohtien rekonstruointi voi olla myöhemmin haastava tehtävä ja lopputuloksen luotettavuus voidaan kyseenalaistaa (Chang 2008, 72.) Chang (2008, 88) kehottaa yhdistämään muistoista kerättävän datan muuhun dataan, jotta voidaan luoda autoetnografinen aineisto analyysia ja tulkintaa varten.

Tähän työhön keräämäni muistot koostuvat puhelimella otetuista näyttötallennuskuvista sekä -videoista, pelin vanhemmista versioista, sekalaisista muistiinpanoista sekä vuosina 2017-2018 ajalla pitämistäni päiväkirjoista. Havainnoinnilla kerätty tuore aineisto on opinnäytetyöni pääosassa, mutta muistin varassa oleva aineisto on pohja työlle, koska muisti ja konkreettiset merkinnät menneisyydestä vaikuttavat ajatteluuni tässä hetkessä ja siten havainnointiin.

Muistojen järjestelmälliseksi keräämiseksi toteutin aikajanan, joka keskittyi menneisyydessä löytämiini skaalautumisen ongelmiin. Aluksi kävin läpi päiväkirjojen avulla työtehtäviäni, ja muistelin mitkä niistä liittyivät skaalaukseen, jos se ei selvinnyt suoraa tekstistä. Päiväkirjoista sain tehtäville päivämäärät, joten pystyin yhdistämään tehtäviä mahdollisiin kuviin ja videoihin sekä etsimään mahdollisesti sen peliversion, missä kyseinen tehtävä oli suoritettu. Näin pystyin vertaamaan peliversiota, jossa skaalautumisen ongelma esiintyi siihen versioon, missä ongelma oli ratkaistu. Tällä tavalla sain näkyväksi ongelman lähtötilanteen ja lopputuloksen.

Olen kirjoittanut ylös satunnaisesti skaalautumiseen liittyviä ratkaisuja, jotta ne eivät pääsisi unohtumaan. Valitettavasti en kuitenkaan muistiinpanoja tehdessäni ole useinkaan kirjoittanut ylös päivämäärää tai kertonut tarkemmin, mihin tämä muistiinpano liittyy. Tästä kokemuksesta viisastuneena aion jatkossa kirjoittaa nämä asiat päiväkirjaani, jolloin tarpeen tullen on helpompaa tietää edes muistiinpanon ajankohta, ja parhaimmillaan pystyn näkemään koko kehityskaaren ongelmasta ratkaisuun.



Kuvio 2. Kohta aikajanalta, jossa on yhdistetty päiväkirjamerkintä muistiinpanoihin ja näyttötallennuskuvaan

Kuviossa 2 on nähtävillä aikajanasta kohta, jolle sain sijoitettua päivämäärän, skaalautumisen ongelman sekä ratkaisun. Tällaisia eheitä muistojen kokonaisuuksia syntyi lopulta

vain muutama, koska kaikkia aineistosta löytyneitä ongelmia en pystynyt yhdistämään tiettyyn päivämäärään saati sitten muistiinpanoon. Tätä aikajanaa käytin kuitenkin hyödykseni havainnoinnin aikana, kun ongelman ilmetessä katsoin janalta, olenko menneisyydessä ratkaisut vastaavaa ongelmaa, ja siten muistot tukivat havainnointia.

3.3 Havainnointiaineisto

Havainnointiaineiston kerääminen tapahtui kahdessa syklissä. Ensimmäinen havainnoinnin sykli oli 19.11.–14.12.2018 ja toinen 7.1.–1.2.2019. Havainnoinnin aikana pyrin skaalautumisen ongelman ilmetessä kokoamaan ongelman tiedot post-it lapuille. Valitsin raakadatan keräämiseen laput siksi, että ne kulkivat tietokonetta paremmin mukanani kaikkialle, ja saatoin kirjoittamisen lisäksi piirtää lappuihin esimerkkikuvia. Merkitsin lappuihin ylös päiväyksen, henkilöt, joiden kanssa ongelmaa kävin läpi, ongelman kuvauksen, mahdollisen hypoteesin ongelman ratkaisemiseksi sekä lopullisen ratkaisun, jos ratkaisu löytyi. Havainnoinnin aikana ja jälkeenpäin kokosin näistä lapuista taulukon, jota sitten käytin analysoinnissa avukseni. Analysoinnin aikana pyrin muun muassa selvittämään taulukon avulla, että oliko saman tyyppisiin ongelmiin ollut useampia ratkaisuja ja mitkä ongelmista kuuluivat samaan kategoriaan.

Alkuperäisestä havainnointiaineiston taulukosta syntyi kaksi muuta taulukkoa, joissa jäsentelin taulukon aineistoa erilaisin teemoin. Yhdessä taulukossa jaoin ongelmat Unity UI:n työkalujen perusteella, ja toisessa jaoin ongelmat niiden laadun perusteella. Ensimmäinen taulukko toimi analysoinnin aikana välineenä sille, että pystyin katsomaan taulukosta, mitkä työkalut olivat olleet osallisena ongelman ratkaisuun. Toisen taulukon avulla sain ongelmille selkeät teemat, jotka osoittautuivat hyödyllisiksi ongelmien ja ratkaisujen jäsentelyssä.

3.4 Haastattelu

Kuten selitin autoetnografista haastattelua käsittelevässä luvussa, haastattelin opinnäytetyöni tiimoilta mentoriani, Jani Reinikaista. Haastattelu tapahtui 9.4.2019 ja litteroin haastattelun aikana nauhoitetun äänitteen kevyesti. Litteroinnin jälkeen etsin tekstistä teemat, joiden mukaan otsikoin haastattelun eri vaiheet. Lopuksi kävin läpi opinnäytetyöni litteroinnin teemojen mukaan, ja Expand-asetusta, ankkurointijärjestelmää ja Layout Element -komponenttia koskevat kohdat saivat lisää informaatiota. Otin myös Layout Element -komponentista pois paikkansa pitämätöntä tietoa.

Lisäsin työhöni lähes kaikki Reinikaisen antamat parannusehdotukset, mutta jätin työn ulkopuolelle iPadin resoluutiota koskevan lisäyksen. Reinikainen (haastattelu 9.4.2019) ehdotti käyttämään skaalauksen vaikutuksien havainnollistamisessa iPadin resoluutiota mobiililaitteiden ohella. Koen, että tämä ehdotus on oikein hyvä ja varmasti selventäisi asiaa aiheeseen perehtymättömälle lukijalle, mutta rajaan iPadeja koskevan skaalautumisen tämän työn ulkopuolelle. Teen rajauksen siksi, että The Walking Dead: Our World -mobiilipelin käynnistäminen iPadilla ei näytä todellista iPadilla skaalautumista, koska peli avataan iPadilla mobiililaitteen resoluutiossa lisäämällä vain tummat reunat pelin ympärille. Saadakseni iPadilla tapahtuvan skaalautumisen näkyväksi minun pitäisi tehdä skaalaus Unityssä, eikä tätä skaalausta voisi toisintaa enää itse laitteella. Eli ulkopuolinen ei voisi nähdä samaa skaalausta. Koen, että tablettaja koskevasta skaalauksesta voisi jatkaa tulevaisuudessa vaikkapa toisessa opinnäytetyössä.

4 Unity UI

Unity on Unity Technologies -yrityksen kehittämä monialustainen pelimoottori, jolla voidaan kehittää kaksi- ja kolmiulotteisia selain-, konsoli-, tietokone- ja mobiilipelejä. (Unity Technologies 2018.) Unity UI taas on Unityn kehittämä käyttöliittymäsuunnittelun työkalupakki. Unity UI on sisäänrakennettu Unityyn ja sisältää oleellisimmat käyttöliittymään tarvittavat elementit kuten napit, listat, kuvat ja tekstit. Unity UI:n osaksi määritellään myös näiden elementtien toiminnallisuudet ja ominaisuudet. (Godbold 2018, 8.)

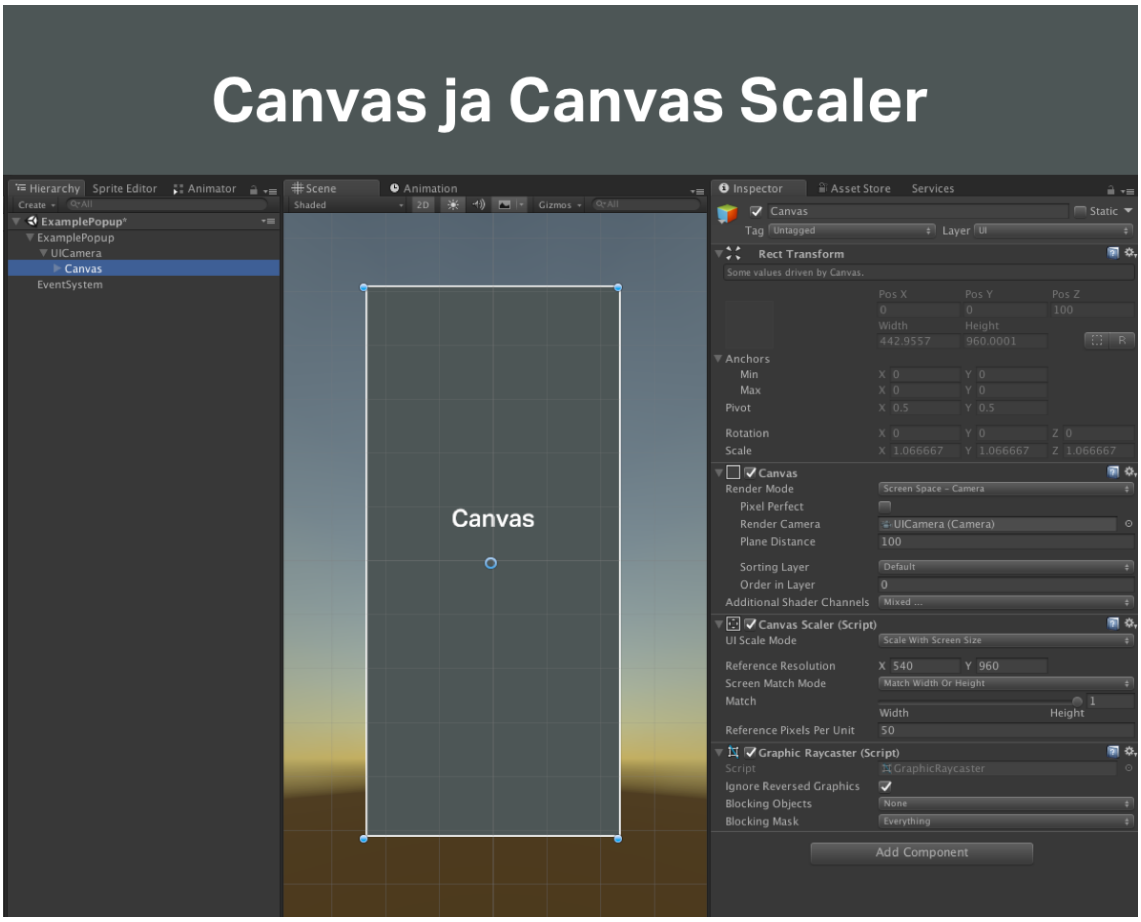
Jacksonin (2015, johdanto) mukaan moni ei uskonut, että Unity tulisi lunastamaan lupauksiaan uusista käyttöliittymätyökaluista, kun Unity Technologies ilmoitti kehittävänsä Unity UI:n. Vähättelevään suhtautumiseen myötävaikutti se, että kehitystyö kesti vuosia jatkuvien viivytyksien vuoksi. Ajateltiin jopa, ettei uutta versiota tulla koskaan näkemään. (Jackson 2015, johdanto.) Kuitenkin Unity UI näki lopulta päivänvalon vuonna 2014 (Fine 2015). Jo ennen julkaisuaan yksi Unity UI:n tärkeimmistä uusista toiminnoista ja työnkulkua edistävistä ominaisuuksista oli *Canvas*-komponentti sekä UI elementtien ankkurointijärjestelmä (Johansen 2014). Canvas-komponentti ja ankkurointijärjestelmä ovat skaalautuvan käyttöliittymäsuunnittelun tukipilareita. Näiden kahden ominaisuuden lisäksi *Automatic Layout Groups* -järjestelmä on osoittautunut hyödylliseksi skaalauksessa, sillä järjestelmään kuuluvat komponentit mahdollistavat automaattisen asemoinnin usealle elementille yhtä aikaa. Vanhassa järjestelmässä vastaavaa automatiikkaa ei ollut ollen-

kaan (Jackson 2015, luku 1, kappale ”Groups”). Layout Element -komponentti taas mahdollistaa yksittäisten elementtien hallinnan silloin, kun elementti on osa Automatic Layout Groups -järjestelmää. Uuden Unity UI:n tuomat työkalut ovat erittäin oleellisia mobiilipelien kehittämisen kannalta. Tietämättään tai tietoisesti Unityn kehittäjät loivat Unity UI:n työkaluilla mahdollisuuden tehdä skaalautuvaa käyttöliittymää.

Edellä mainituista Unity UI:n työkaluista on tärkeää huomioida se, että ne ovat nimenomaan käyttöliittymän sommitteluun ja hallintaan tarkoitettuja, visuaalisesti näkymättömiä komponentteja. Canvas, ankkurointijärjestelmä, Automatic Layout Groups -järjestelmä ja Layout Element -komponentti ovat skaalautumiselle oleellisia, mutta käyttäjälle ne eivät enää lopputuotteessa näy. Käyttäjä voi lähinnä havaita komponenttien puutteen siinä vaiheessa, kun näkyvässä roolissa olevat visuaaliset elementit käyttäytyvät mobiililaitteella kummallisella tavalla. Näiden näkymättömien komponenttien selkeyttämiseksi käytän esimerkkikuvissa komponenttien erotteluun eri värisiä kuvioita. Huomioitavaa on, että näitä kuvioita ei ole käytössä Unity UI:ssa vaan ne ovat puhtaasti tämän työn seuraamisen helpottamista varten tehty visuaalinen kirjasto. Lisäksi käytän tekstissä samoja värejä komponenttien yhteydessä. Tämä ei ole perinteistä opinnäytetöissä, mutta uskon sen helpottavan työn seuraamista.

4.1 Canvas ja Canvas Scaler

Canvas-elementti on englanninkielisen nimensä mukaisesti maalauspohja, johon käyttöliittymän elementit piirretään (Jackson, 2015, luku 2, kappale ”The Canvas”). Canvas mahdollistaa kolmiulotteisen käyttöliittymän suunnittelun, mikä oli mullistavaa komponentin ilmestyessä vuonna 2014. Canvas-elementtiin on sisäänrakennettu resoluution skaalauksen työkalu, jota kutsutaan Canvas Scaler -komponentiksi (Jackson 2015, luku 2, kappale ”Resolution and scaling”). Keskityn siksi tässä luvussa pääsääntöisesti Canvas Scaler -komponenttiin, koska sen eri asetuksilla säädetään käyttöliittymän skaalautumista. Kuviossa 3 on esillä Canvas-elementin asetukset ja myös Canvas Scaler -komponentti on nähtävillä.



Kuvio 3. Canvas elementti, jossa Canvas Scaler -komponentti näkyvissä

Canvas Scaler -komponentilla on muutamia eri asetuksia, joiden mukaan komponentti kontrolloi käyttöliittymää. Yksi asetuksista on *UI Scale Mode*, jonka voisi löyhästi suomentaa UI:n skaalauksen tilaksi. Käytännössä tämä asetus määrittää, miten komponentin sisällä olevat UI-elementit skaalataan. Tälle asetukselle on mahdollista määritellä tarkemmin tila kolmesta eri vaihtoehdosta. *Scale with Screen Size* -tila valitaan silloin, kun käyttöliittymän halutaan skaalautuvan näytön koon mukaan. Kaksi muuta asetusta eivät skaalaa käyttöliittymää oikeastaan lainkaan, koska *Constant Pixel Size* pitää elementtien pikselikoon aina vakiona, ja *Constant Physical Size* taas pitää elementtien fyysisen koon aina vakiona.

Kun näytön koon mukaan skaalautuminen on valittuna, avautuu Canvas Scaler-komponenttiin lisäasetuksia, joilla voidaan määrittää miten skaalautuminen tapahtuu. *Match Width or Height* -asetus, eli sovittaminen leveyteen tai korkeuteen, on asetus, joka nimensä mukaan skaalaa käyttöliittymää joko leveyden tai korkeuden mukaan. *Expand*-asetus toimii siten, että jos näytön koko on pienempi kuin käyttöliittymän natiiviresoluutio,

käyttöliittymää levitetään sopimaan natiiviresoluutioon. *Shrink*-asetus taas toimii niin, että jos näyttö on suurempi kuin natiiviresoluutio, käyttöliittymää kutistetaan, kunnes käyttöliittymä vastaa natiiviresoluutiota. (Godbold 2018, 38–42.) Expand- ja Shrink-asetuksilla ei ole omia lisäasetuksia, koska ne toimivat automaattisesti tiettyjen valmiiden sääntöjen mukaan (Godbold 2018, 40–41).

Jackson (2015, luku 2, kappale ”Scale with Screen Size”) selittää Expand-asetusta niin, että asetus skaalaa käyttöliittymää “alaspäin” ja lisää etäisyyksiä elementtien välille varmistaakseen, että natiiviresoluution käyttöliittymä piirtyy skaalattavaan koon sisällä aina natiiviresoluutiossa. Käytännössä tämä asetus saattaa aiheuttaa esimerkiksi sitä, että tekstin koko pienenee siirryttäessä kapeampaan kuvasuhteeseen. Haastatellessani Jani Reinikaista (haastattelu 9.4.2019) aiheesta hän selitti Expand-asetuksen toimintaa vähän toisella tavalla. Kenties käännöstyössä informaatiota menetetään tai sitten, Reinikaista (haastattelu 9.4.2019) lainatakseni, kukaan ei oikein osaa selittää Expand-asetuksen toimintaperiaatteita. Reinikainen (haastattelu 9.4.2019) kertoi puhtaasti testaamalla selvittäneensä, miten Expand-asetus toimii.

Reinikaisen (haastattelu 9.4.2019) mukaan Expand-asetus ensin sovittaa natiiviresoluution käyttöliittymän skaalattavan resoluution sisään. Natiiviresoluutio siis täyttää uuden resoluution pinta-alan joko leveys- tai korkeussuunnassa. Tämän jälkeen natiiviresoluution ympärille voi jäädä tyhjää tilaa, joten Canvas vielä levitetään jäljelle jääneeseen tilaan. (Reinikainen, haastattelu 9.4.2019.) Reinikaisen (haastattelu 9.4.2019) mukaan suunnittelijan näkökulmasta Expand-asetusta käytettäessä natiiviresoluution pinta-ala on aina käytettävissä. Skaalattavasta resoluutiosta riippuen, pinta-ala voi vain saada lisää tilaa joko leveys- tai korkeussuunnassa, koska Canvas ei voi koskaan jäädä vajaaksi. Tilaa tulee lisää myös vain joko leveys tai korkeussuunnassa, ei koskaan molemmissa. Expand-asetus ei myöskään millään tavalla vaikuta yksittäisiin elementteihin vaan kaikki mitä yksittäisille elementeille tapahtuu skaalatessa, on vain seurausta siitä, mitä Canvakselle tapahtuu. (Reinikainen, haastattelu 9.4.2019.) Alla oleva kuvio 4 mukailee Reinikaisen (haastattelu 9.4.2019) piirtämää esimerkkikuvaa Expand-asetuksen toimintaperiaatteesta.

Kuvio 4. Expand-asetuksen toimintaperiaate Reinikaisen piirrosta mukaillen

Shrink-asetus taas on hieman outo, mutta ehkä hyödyllinen joissain tilanteissa, sillä se skaalaa käyttöliittymää ylöspäin ja rajaa sen sitten sopimaan näytölle (Jackson 2015,

luku 2, kappale ”Scale with Screen Size”). Tämä on ongelmallista lähinnä siksi, että suunnittelijan on vaikeaa ennakoida, miten ja mistä rajaus tapahtuu.

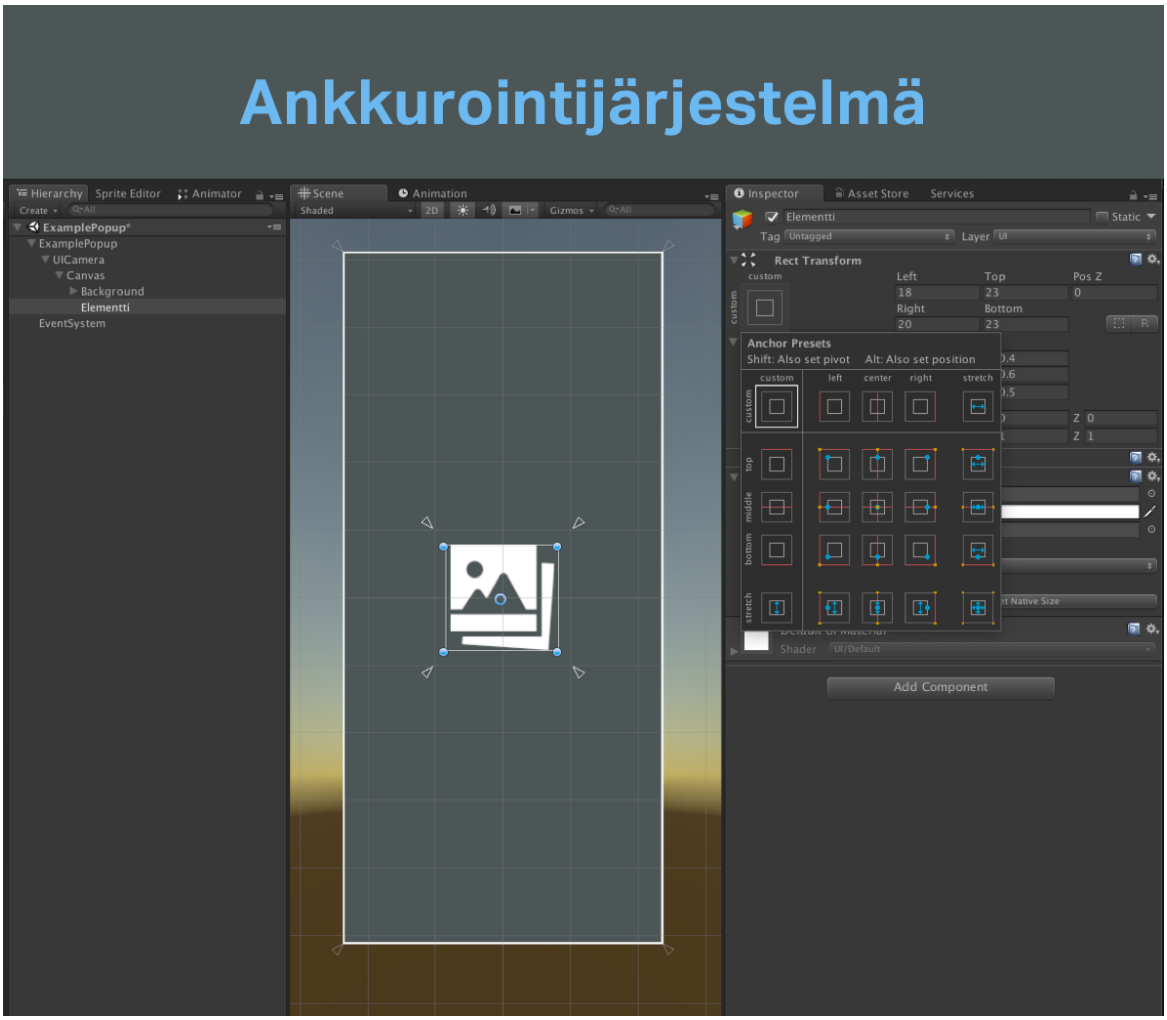
Match Width or Height -asetus taas tarjoaa lisäasetuksena mahdollisuuden tarkentaa, haluaako käyttöliittymän pyrkivän suosimaan skaalatessa leveyttä, korkeutta vai jotain siltä väliltä (Jackson 2015, luku 2, kappale ”Scale with Screen Size”). Käytännössä tämä tapahtuu niin, että asetuksessa voi valita arvon numeron 0 ja 1 väliltä. Jos arvo on lähempänä nollaa, skaalaus suosii leveyttä ja jos numero on lähempänä yhtä, skaalaus suosii korkeutta. Jos arvo on 0.5, asetus pyrkii säilyttämään elementtien etäisyydet sekä leveys- että korkeussuunnassa. (Godbold 2018, 40–41.) Tavallaan siinä, missä Expand ja Shrink ainoastaan joko laajentavat tai kutistavat käyttöliittymää, Match Width or Height -asetus voi tehdä molempia yhtä aikaa. Käyttöliittymän elementit saattavat siis liikkua arvaamattomasti, mutta skaalaus on dynaamisempi, sillä elementtejä ei vain laajenneta tai kutisteta mielivaltaisesti vaan suunnittelijalla on mahdollisuus tasapainottaa skaalausta.

Valitettavasti asetukset eivät kuitenkaan automaattisesti sovi kaikille käyttöliittymille tai kaikkien laitteiden kuvasuhteille. Siksi valitsemisen pitäisi perustua siihen, miten suunnittelija haluaa käyttöliittymän skaalautuvan, ja millaisesta käyttöliittymästä on kyse. (Godbold 2018, 40–41.)

4.2 Ankkurointijärjestelmä

Tässä luvussa käsittelen ankkurointijärjestelmää, joka mahdollistaa käyttöliittymän elementtien ankkuroinnin suhteessa Canvakseen. Esimerkiksi oikeaan yläkulmaan ankkuroitu elementti on skaalauksesta huolimatta aina oikeassa yläkulmassa. Tämä voi kuulostaa itsestään selvältä, mutta koska elementtejä on mahdollista raahata vapaasti paikasta toiseen ruudulla, suunnittelijat saattavat unohtaa ankkuroinnin, ja sitten skaalatessa elementit liikkuvat odottamattomiin paikkoihin. Kuviossa 5 on nähtävillä esimerkki ankkuroinnista kuvallisen elementin ympärillä.

-

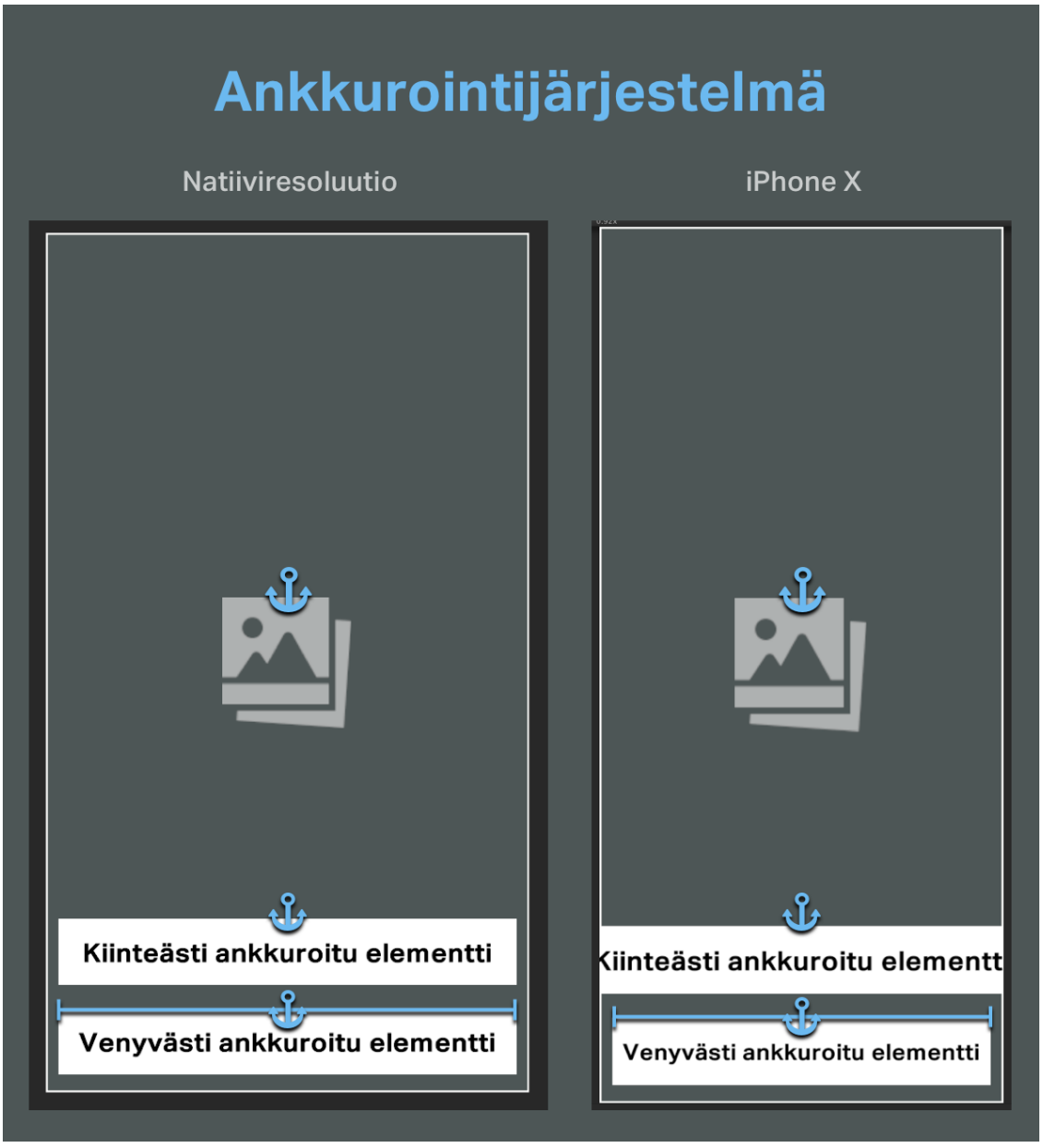


Kuvio 5. Ankkuroinnin kahvat ja nivelpiste sekä elementin ankkurointiin käytettävät Anchor Presets -asetukset

Kaikilla käyttöliittymän elementeillä on valmiiksi kulmissaan kahvat sekä keskellä nivelpiste, joiden avulla elementti asemoidaan ja skaalataan halutulla tavalla. (Godbold 2018, 50) Kahvoilla määritellään elementin ankkuripisteet suhteessa näyttöön, ja nivelpisteellä taas määritellään mistä kohtaa elementti skaalaa ja kiertää itsensä ympäri.

Ankkurointijärjestelmä on hyödyllinen skaalautuvaa käyttöliittymää tehdessä myös siksi, että ankkurointi voidaan asettaa joko kiinteäksi tai joustavaksi. Ankkuroinnin ollessa kiinteä, elementti ei huomioi lainkaan Canvaksen skaalautumista vaan pitää kokonsa. Ankkuroinnin ollessa joustava elementti huomioi Canvaksen muutoksen ja venyy tai kutistuu eli skaalaa Canvaksen sisällä. Kun elementti on määritelty joustavaksi, elementin ympärille on mahdollista lisätä marginaaleja ja esimerkiksi määritellä, että elementin leveys on

aina 50 prosenttia näytön leveydestä. Ankkuroinnilla on siis mahdollista tehdä dynaamisia ja responsiivisia käyttöliittymiä, jotka mukautuvat resoluutioon juuri kuten suunnittelija haluaa. (Jackson 2015, luku 4, kappale ”Stretch it, bend it”.)



Kuvio 6. Kiinteän ja joustavan ankkuroinnin ero skaalatessa

Kuviosta 6 näkee, että kiinteästi ankkuroitu teksti rajautuu reunasta. Venyvästi ankkuroitu teksti taas mukautuu uuteen leveyteen.

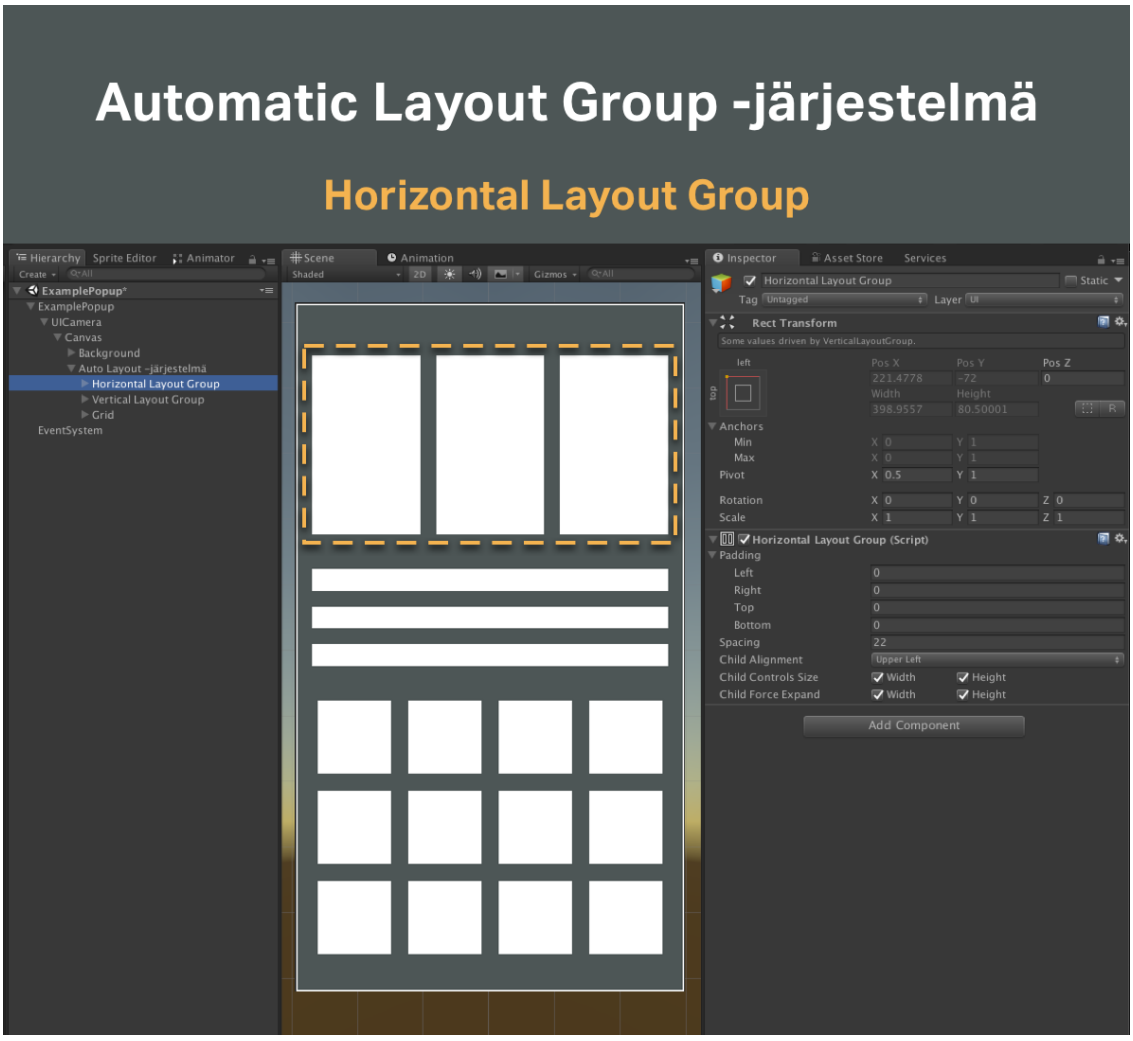
Reinikainen (haastattelu 9.4.2019) sanoi, että hän ajattelee usein käyttöliittymään liittyviä asioita printtipuolen näkökulmasta. Näin ollen Reinikaisen (haastattelu 9.4.2019) mukaan ankkuroinnissa ei ole kyse niinkään siitä, miten elementit siirtyvät ruudulla, vaan ankkuroinnilla pystytään määrittelemään johdonmukainen rakenne käyttöliittymälle eli luomaan esimerkiksi marginaaleja ja palstoitusta. Ankkurointijärjestelmässä on tiettyjä samoja piirteitä kuin verkkosivujen puolella käytettävässä CSS-koodissa, mutta ankkurointi ei suoranaisesti toimi samalla tavalla (Reinikainen, haastattelu 9.4.2019).

4.3 Auto Layout -järjestelmä

Tämän luvun aiheena on Automatic Layout Groups -järjestelmä sekä ennen kaikkea järjestelmään kuuluvat komponentit. Automatic Layout Groups -järjestelmä on oleellinen silloin, kun käyttöliittymän elementtien määrä ei ole vakio. Esimerkki tästä voisi olla peleissä usein käytössä oleva pelaajan inventaario, johon pelaajan löytämät esineet kerätään pelin edetessä. Elementtien määrä siis vaihtelee, mutta kaikki elementit pitäisi olla mahdollista sisällyttää käyttöliittymään. Tässä kohtaa Automatic Layout Groups -järjestelmä astuu kuvaan. Järjestelmän komponenteilla voidaan rivittää, asemoida ja skaalata elementit määriteltyjen asetusten mukaan, joten jokaista yksittäistä elementtiä ei tarvitse asemoida erikseen. (Godbold 2018, 79.)

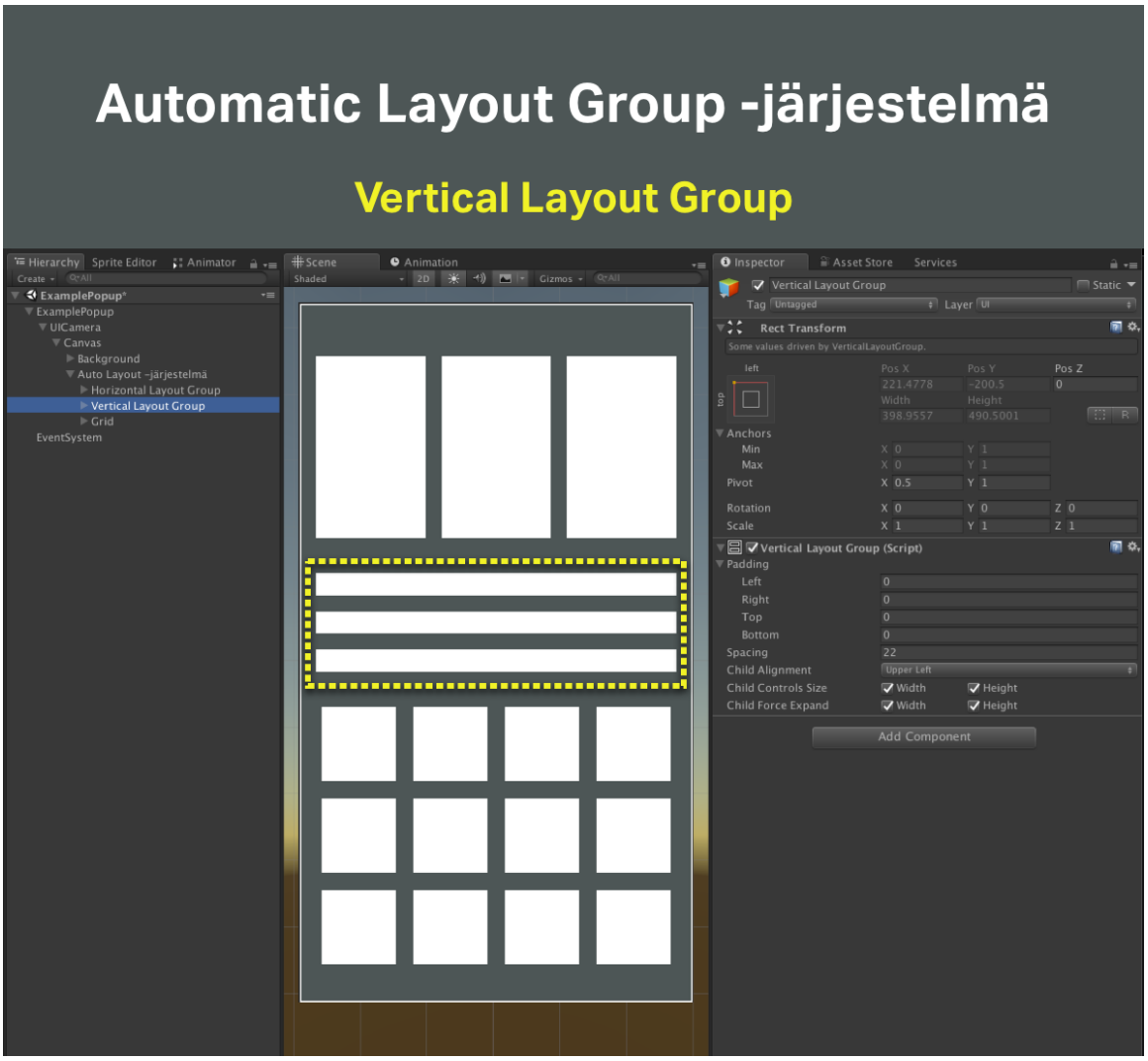
Automatic Layout Groups -järjestelmän komponenteilla voi määritellä marginaaleja, jotka toistuvat kaikissa elementeissä. Olen huomannut, että Automatic Layout Groups -järjestelmää kannattaakin käyttää tämän ominaisuuden vuoksi myös ihan vain ulkoasun sommitteluun myös silloin, kun elementtien määrä on vakio. Ankkuroinnilla tehtyyn palstoitukseen verrattuna, Automatic Layout Group -järjestelmän komponentteja käytettäessä elementit ovat tietoisia toistensa olemassa olosta. Jos esimerkiksi yksi elementti piilotetaan käyttöliittymästä, muut elementit voivat täyttää jäljelle jäneen tilan. Ankkuroinnilla elementit pitävät ennalta määrätyt sijaintinsa, eivätkä toimi yhtä dynaamisesti.

Automatic Layout Groups -järjestelmän komponentteja on kolme erilaista. Vaakatasossa ryhmittelevä eli Horizontal Layout Group -komponentti, pystysuunnassa ryhmittelevä Vertical Layout Group -komponentti sekä ruudukon muodossa ryhmittelevä Grid Layout Group -komponentti. Kuvioissa 7–9 on esimerkit näiden komponenttien toiminnasta.



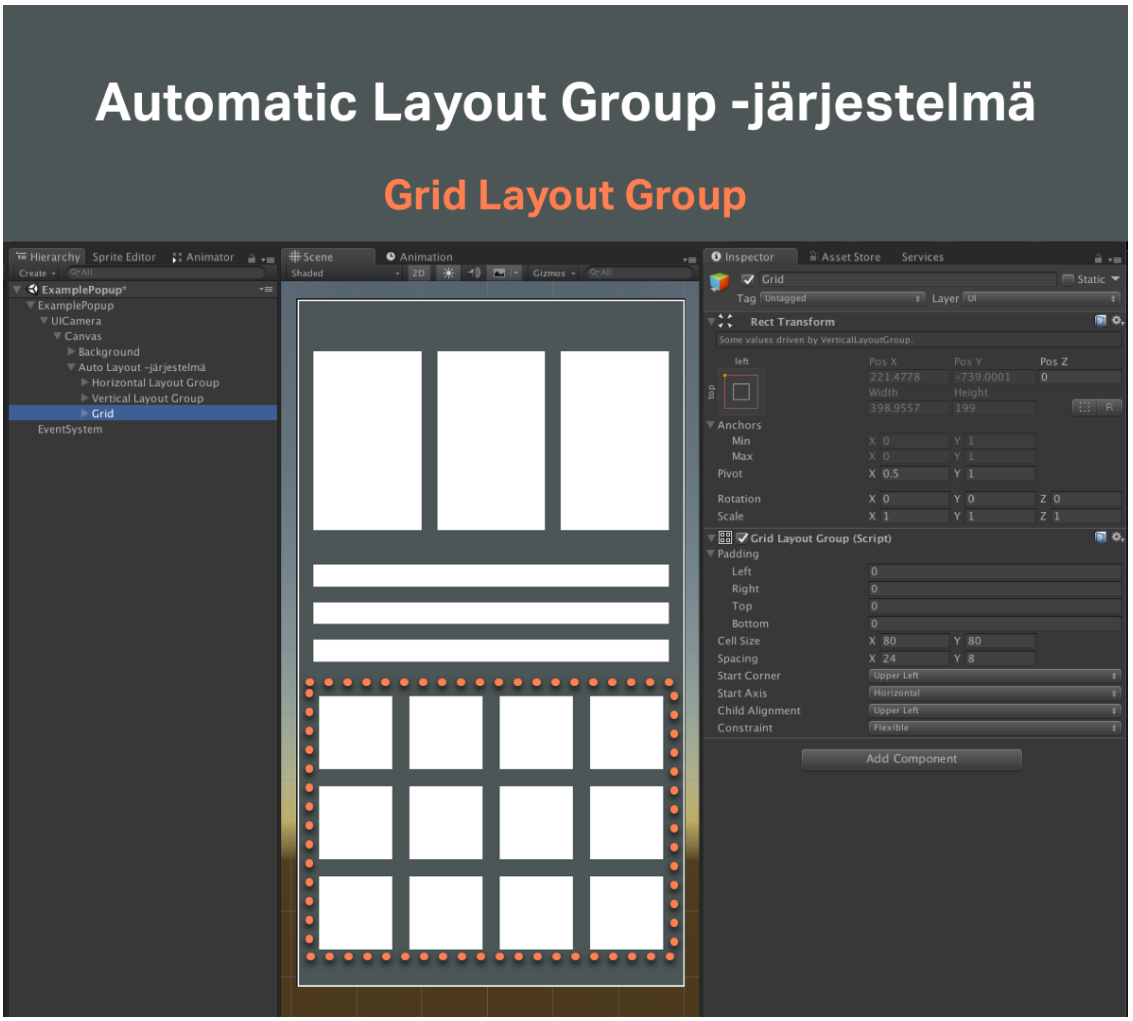
Kuvio 7. Automatic Layout Groups -järjestelmän Horizontal Layout Group -komponentti sekä esimerkki ryhmittelystä

Horizontal Layout Group -komponentti alle sijoitetut elementit asettuvat automaattisesti rinnakkain. Tässä työssä tätä komponenttia kuvastaa oranssi, suorakaiteista koostuva katkoviiva.



Kuvio 8. Automatic Layout Groups -järjestelmän Vertical Layout Group -komponentti sekä esimerkki ryhmittelystä

Vertical Layout Group -komponentti alle sijoitetut elementit taas asettuvat automaattisesti allekkain. Tässä työssä tätä komponenttia kuvastaa keltainen, neliöistä koostuva katkoviiva.



Kuvio 9. Automatic Layout Groups -järjestelmän Grid Layout Group -komponentti sekä esimerkki ryhmittelystä

Grid Layout Group -komponentti alle sijoitetut elementit taas asettuvat automaattisesti allekkain. Tässä työssä tätä komponenttia kuvastaa punainen, ympyröistä koostuva katkoviiva. Toisin kuin Horizontal Layout Group- ja Vertical Layout Group -komponenteissa, ruudukkoa varten täytyy määritellä kaikille elementeille yksi korkeus ja leveys, koska tämän perusteella ruudukko sitten muodostetaan. Grid Layout Group -komponentilla ei siis voi luoda dynaamista ruudukkoa, jossa elementtien koot vaihtelisivat.

4.4 Layout Element

Layout Element -komponentilla voidaan uudelleen määritellä koko elementeille, joiden kokoa muuten määrittelisi Automatic Layout Group -komponentti. Jos Automatic Layout

Group -komponentti yrittää muuttaa elementin kokoa, Layout Element -komponentti yli-ajaa omilla asetuksillaan Automatic Layout Group -komponentin lähettämät määritelmät. (Godbold 2018, 93.)

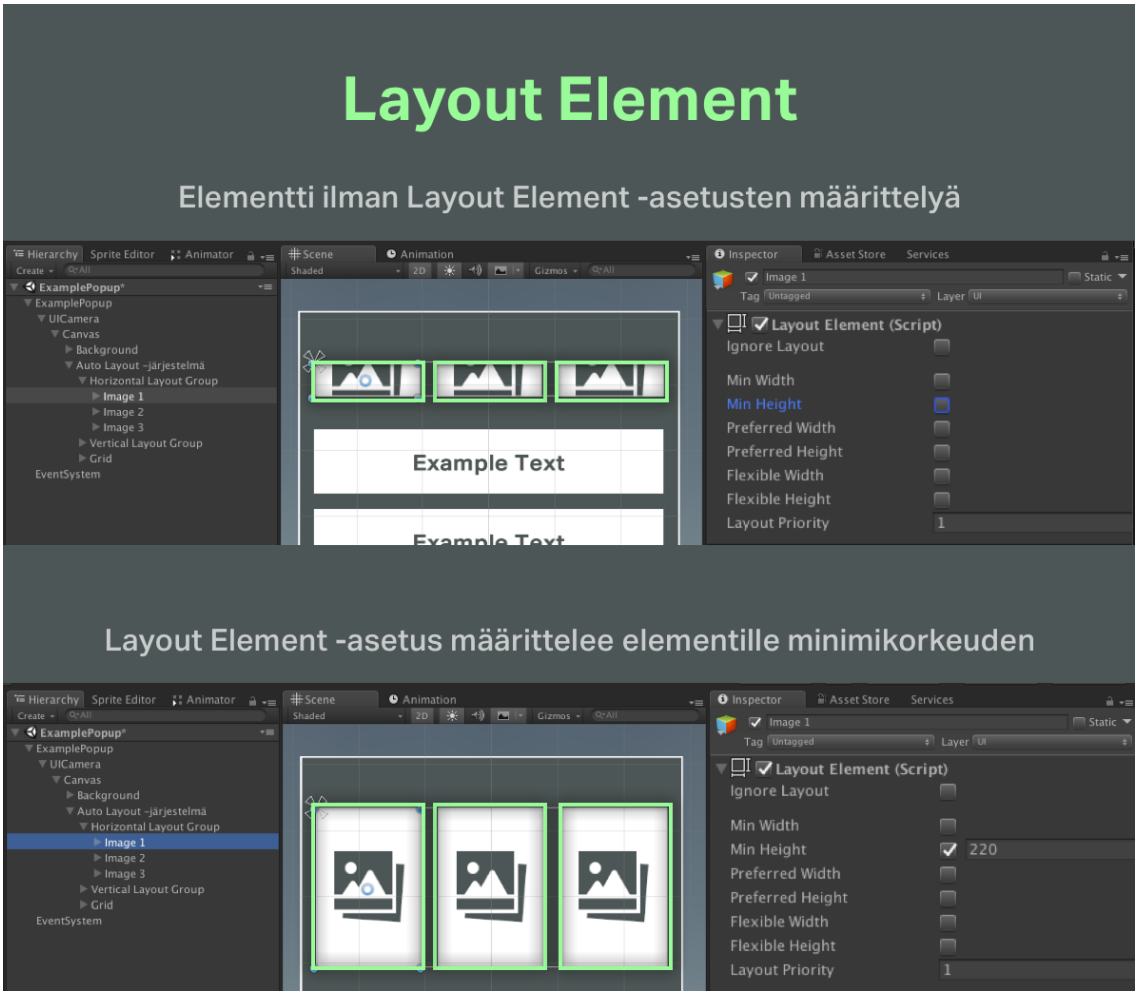
Sapion (2015) mukaan ymmärtääkseen paremmin Layout Element -komponentin toimintaa täytyy huomioida, että jokainen käyttöliittymän elementti on periaatteessa Layout Element. Elementit ovat tietoisia omasta koostaan mutta ne eivät voi kontrolloida omaa kokoaan suoraan (Sapio 2015, 24). Reinikainen (haastattelu 9.4.2019) selitti, että hierarkia on erittäin keskeisessä roolissa, kun puhutaan Layout Element -komponenteista. Reinikaisen mukaan tärkeintä on ymmärtää hierarkian yhteys, sillä jokainen askel hierarkiassa ylöspäin tai alaspäin tarvitsee selvää kommunikaatiota elementtien välillä, mutta jokainen tapaus on vähän omanlaisensa.

Reinikaisen mukaan on tapauksia, joissa on mielekkäämpää rakentaa kokonaisuudet niin, että automatic layout group -komponentin alle tehdään vain yksi rivi, jolle annetaan mitat Layout Element -komponentilla. Tämän Layout Element -komponentin sisältämän rivin sisälle voidaan sijoittaa sitten elementtejä ihan vain ankkuroinnilla, ja näihin sisällä oleviin elementteihin ei enää vaikuta Automatic Layout Group -komponentin asetukset. Reinikaisen mukaan tämä on pelin performanssin kannalta myös parempi ratkaisu, koska taustalla vaikuttavat laskennat loppuvat Layout Element -komponenttiin. Mutta silloin kun elementtien pitää olla dynaamisia, esimerkiksi tekstin pituus määrittää elementin kokoa tai elementtien määrä voi vaihdella, niin on mielekästä käyttää Automatic Layout Group -komponentteja läpi koko hierarkian. (Reinikainen, haastattelu 9.4.2019)

Reinikainen (haastattelu 9.4.2019) selitti myös, että Layout Element -komponentti ei yli-aja varsinaisesti mitään Automatic Layout Group -komponentin asetuksia, vaan ainoastaan tekee näkyväksi jo olemassa olevat arvot ja mahdollistaa manuaalisesti arvojen syöttämisen. Layout Element -komponentti ei itsessään tee mitään vaan se on työkalu, jolla suunnittelija voi antaa itselleen lisää kontrollia.

Koko tämän kuvion komponenttien välillä voisi nähdä eräänlaisena vuoropuheluna. Jos esimerkiksi halutaan, että Automatic Layout Group -komponentin alla olevalle elementille tietty koko, niin elementti tarvitsee Layout Element -komponentin kertoakseen ylöspäin Automatic Layout Group -komponentille omat mittansa, jotka suunnittelija on määritellyt. Layout Element -komponentti mahdollistaa tällaisen kommunikoinnin. Layout Element -komponentti tuo näkyviin suunnittelijalle jo olemassa olevat asetukset, ja Layout Element

-komponentilla suunnittelija antaa itselleen mahdollisuuden kontrolloida jo näitä olemassa olevia asetuksia.



Kuvio 10. Esimerkki Layout Element -komponentin käytöstä elementin koon määrittelyyn

Kuviossa 10 on nähtävissä esimerkki, jossa Horizontal Layout Group -komponentin sisällä olevassa elementissä on Layout Element -komponentti, jossa ensin ei ole koon määrityksiä, ja sitten minimikooksi määritellään 220. En sen tarkemmin käy läpi Layout Element -komponentin asetuksia, mutta oleellista edellisestä kuvasta on huomioida se, että riittää, että vain yhdessä elementissä on Layout Element -komponentti. Kun yksi elementti kertoo minimikokonsa Horizontal Layout Group -komponentille, muut elementit seuraavat perässä.

5 Analyysi

Tässä luvussa käyn läpi havainnoinnin aikana esiintyneitä ongelmia sekä niihin löytyneitä ratkaisuja. Käyn ongelmia läpi esimerkkien avulla, mutta pyrin luvuissa käsittelemään ongelmia yleisesti yksittäisten tapauksien sijaan, jotta tietoa voisi helpommin soveltaa. Ongelmille muodostui analyysivaiheessa kolme teemaa: koko käyttöliittymän skaalautumisen ongelmat, elementtien skaalautumisen ongelmat suhteessa toisiinsa sekä yksittäisen elementin skaalautumisen ongelmat.

Oli helppoa jakaa ongelmat koskemaan joko koko käyttöliittymää tai vain yhtä elementtiä. Ensimmäisen teeman alle tulivat ongelmat, jotka vaikuttivat useampiin elementteihin yhtä aikaa skaalatessa, ja toiseen teemaan kuuluivat ongelmat, joissa vain yksi elementti tuotti ongelmia. Tässä vaiheessa koko käyttöliittymää koskevan teeman alle kasautui huomattavasti enemmän ongelmia. Halusin jakaa tämän teeman ongelmat pienempiin pinoihin, ja analysoidessa havaitsin, että osa ongelmista oli Automatic Layout Groups -järjestelmästä johtuvia, joten siitä tuli kolmas teema. Halusin tosin vielä löytää teemalle selkeämmän nimen. Koska ongelmat juontavat juurensa siitä, että Automatic Layout Groups -järjestelmän sisällä olevat elementit vaikuttavat aina toisiinsa, kolmanneksi teemaksi tuli elementtien skaalautumisen ongelmat suhteessa toisiinsa.

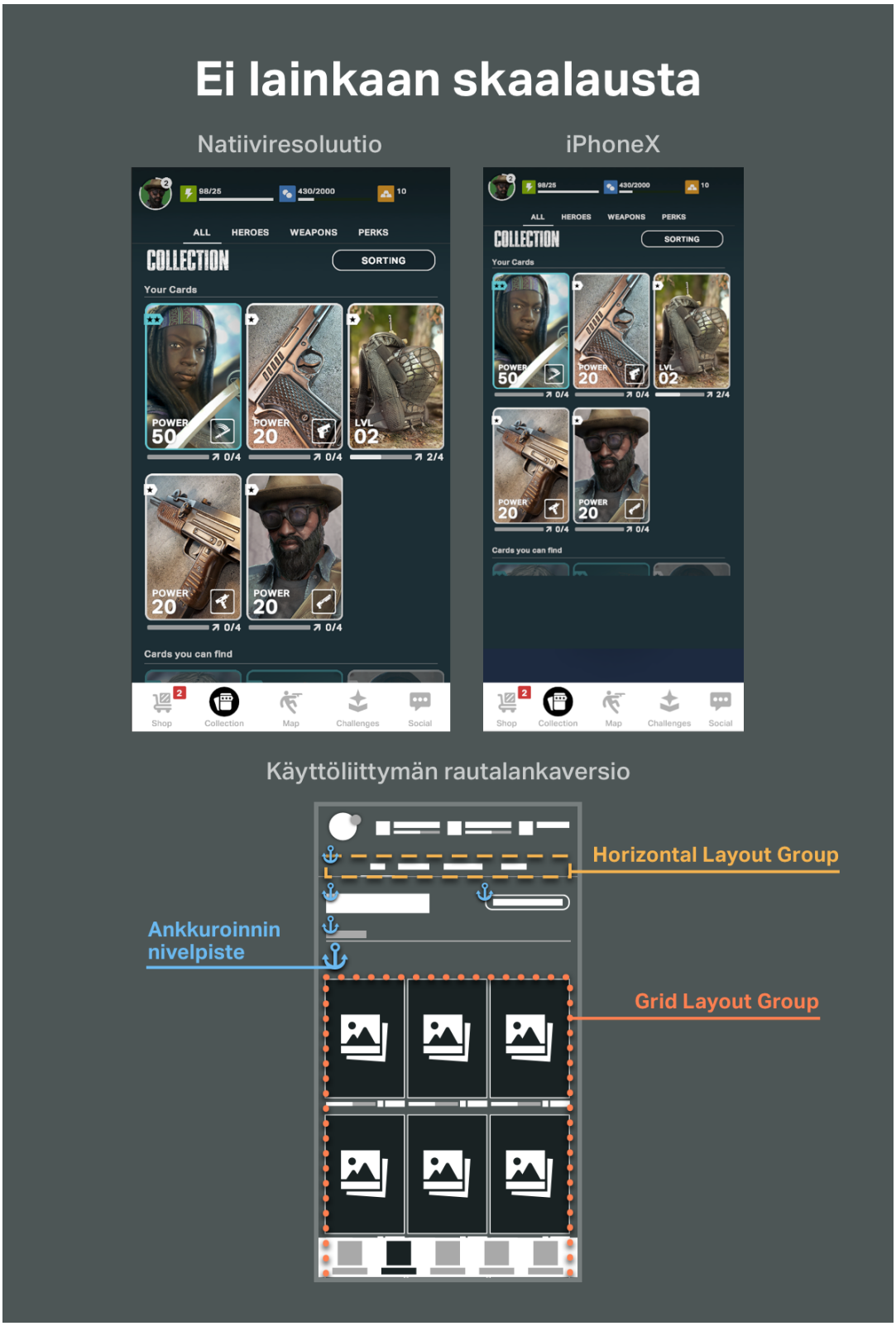
5.1 Koko käyttöliittymän skaalautumisen ongelmat

Tässä luvussa keskitytään ongelmiin, joissa käyttöliittymä kokonaisuutena käyttäytyy oudosti skaalatessa. Tällaisia ongelmia nousi esiin neljä kertaa havainnoinnin aikana. Tässä ja seuraavassa luvussa on hyvä huomioida, että ongelmien ratkaisua lähdetään hakemaan ylhäältä alaspäin. Vaikka ongelma näyttäisi olevan yksittäisessä elementissä, pyritään ensin korjaamaan hierarkiassa ylempänä olevat elementit.

5.1.1 Käyttöliittymä skaalaa ennalta-arvaamattomasti

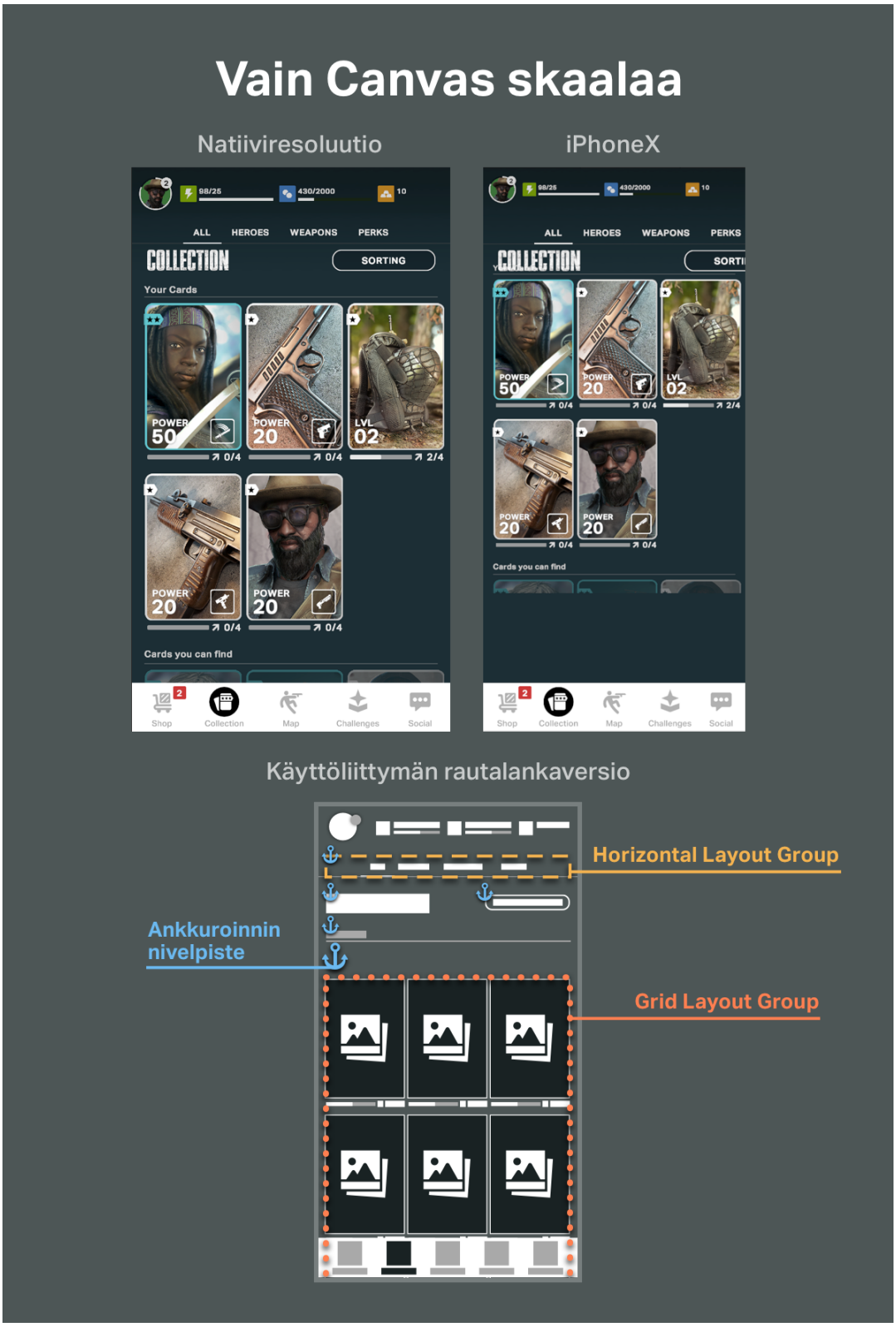
Kaikki skaalautumisen ongelmat ovat periaatteessa ennalta-arvaamatonta käyttöliittymän skaalautumista. Havainnoinnin aikana kuitenkin huomasin tarvitsevani tällaista hieman epämääräistä määritelmää, koska aina ei voinut suoraa sanoa, mistä ongelma juontaa juurensa. Tässä luvussa avaan hieman sitä, miten tällaista tilannetta voi yrittää lähteä purkamaan.

Kuvioista 11–13 voi nähdä, että viimeisessä käyttöliittymässä ulkoasu pysyy lähes yhtenäisenä alkuperäisen natiiviresoluution kanssa. Kahdessa ensimmäisessä kuviossa on nähtävillä miten korttien listaus ei kata koko näyttöä, ja sorting-nappi rajautuu näytön reunassa. Ongelmat johtuvat siitä, että ilman venyvää ankkurointia elementit eivät yksinkertaisesti tiedä, mihin näyttö loppuu. Kun venyvä ankkurointi on käytössä, elementit skaalaavat Canvaksen sisällä näytön reunojen perusteella.



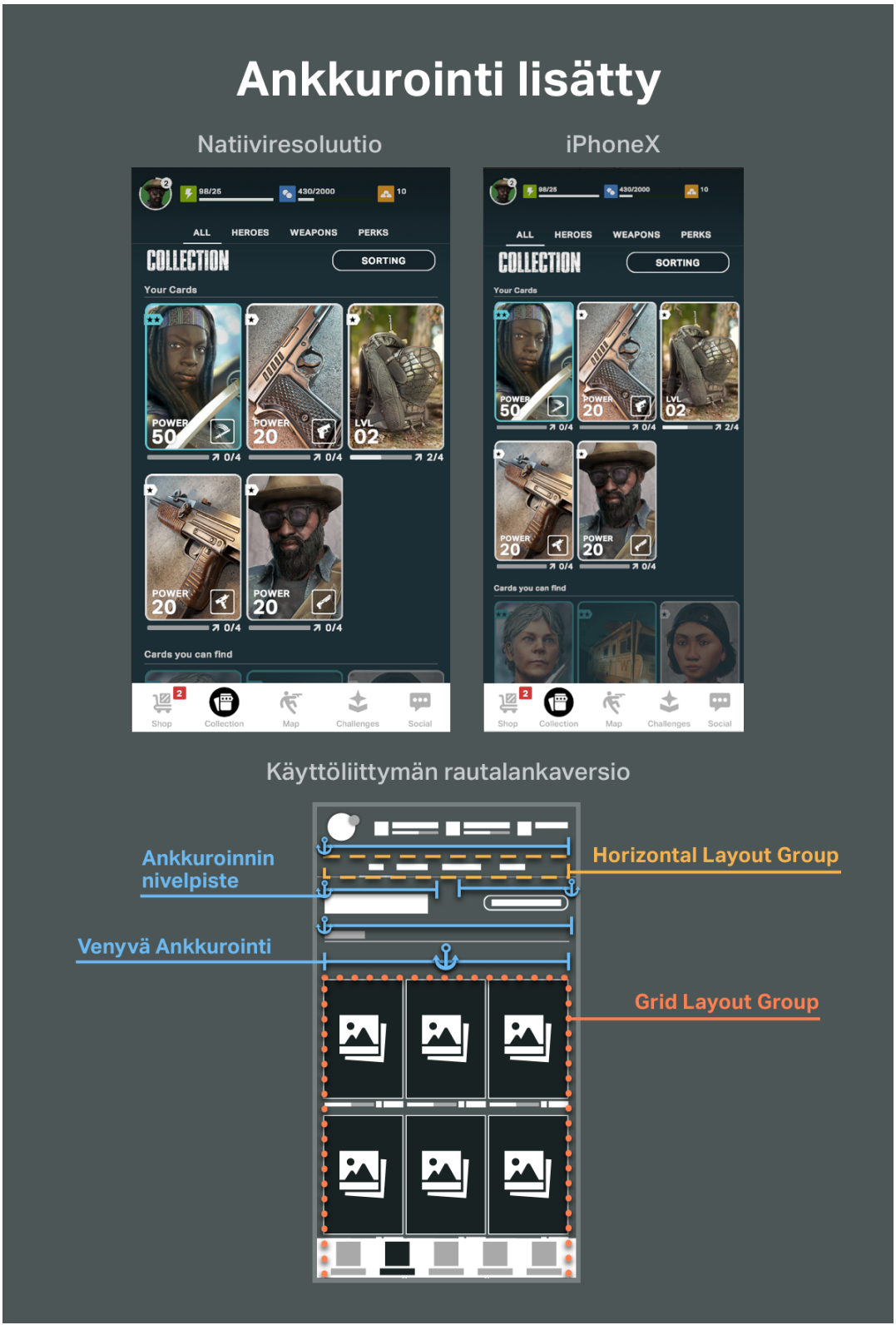
Kuvio 11. Skaalaamaton käyttöliittymä

Kuviossa 11 käyttöliittymässä ei ole lainkaan skaalausta käytössä.



Kuvio 12. Vain Canvas skaalaa käyttöliittymää

Kuviossa 12 vain Canvas huolehtii käyttöliittymän skaalautumisen onnistumisesta.



Kuvio 13. Venyvä ankkurointi lisätty Canvasin skaalauksen lisäksi

Kuviossa 13 on Canvaksen skaalautumisen lisäksi vielä venyvä ankkurointi lähes kaikissa elementeissä.

Olisi voinut olettaa, että ongelma johtuu Canvas scaler -komponentin asetuksista. Ratkaisu skaalaukseen kuitenkin löytyi käyttöliittymän elementtien ankkuroinnin asetuksista. Tällaisia odottamattomia ratkaisuja sattuu yllättävän usein. Ennalta arvaamattomiin sattumuksiin auttaa se, että määrittelee itselleen, miten skaalauksen kuuluu tapahtua. Parempi vielä, jos tekee itselleen mallin skaalauksesta, kuten kuvioissa olevat rautalankaversiot. Määrittelyn ja suunnittelun pohjalta ratkaisu on helpompi löytää. Ylemmässä esimerkissä olisi voitu esimerkiksi määritellä, että käyttöliittymän halutaan skaalatessa pysyvän näytön keskellä ja käyttävän koko näytön koon. Vastaus ongelmaan periaatteessa tuli jo määrittelyssä itsessään; elementtien pitää tietää missä on näytön keskipiste, eli elementtien pitää olla ankkuroituna keskelle, ja lisäksi elementtien pitää tietää näytön koko, mikä on mahdollista venyvällä ankkuroinnilla.

5.1.2 Käyttöliittymän elementit skaalautuvat tarpeettoman suuriksi tai pieniksi

Canvas Scaler -komponentin asetukset saavat skaalatun käyttöliittymän elementit näyttämään joissain tapauksissa isommilta tai pienemmiltä. Elementtien koon muutos johtuu siitä, että skaalaukseen käytettävä logiikka sanelee tietyt ehdot skaalaukselle, ja sen mukaan käyttöliittymää säädellään. Itselleni asia selkeytyi parhaiten, kun kuvittelin, että käyttöliittymän skaalaus rakennetaan joko lego- tai duplo-palikoilla; molemmilla palikoilla saadaan rakennelma aikaiseksi, mutta toinen näyttää sirommalta ja toinen kolhommalta. Koska kumpikaan lopputulos ei vastaa täysin alkuperäistä natiiviresoluutiota, jää suunnittelijan tehtäväksi valita skaalauksen menetelmä visuaalisten seikkojen perusteella.

Expand-asetus pyrkii sovittamaan natiiviresoluutiossa tehdyn käyttöliittymän uuden resoluution sisälle. Reinikainen (haastattelu 9.4.2019) selitti tätä niin, että natiiviresoluutio ensin skaalataan ruudun sisään kokonaisena, ja vasta sen jälkeen Canvas muuttaa muotoaan. Elementtien koko pysyy siis uskollisena natiiviresoluution suhteelliselle koolle, eli elementit vievät saman verran tilaa näytöltä skaalauksen jälkeenkin. Elementtien sijainti vain saattaa muuttua riippuen siitä, miten paljon Expand-asetus joutuu leviämään pysty- tai vaakasuunnassa.

Match Width or Height -asetus pyrkii skaalaamaan käyttöliittymän joko leveyden, korkeuden tai molempien variaatioilla, asetuksista riippuen. Joka tapauksessa asetus yrittää säilyttää elementtien mittasuhteet ja sijainnit, mikä aiheuttaa kapeammilla laitteilla sen, että elementtien välimatkat lyhenevät. Elementit alkavat siten näyttää suuremmilta, koska natiiviresoluution koossa ne vievät enemmän tilaa näytöllä.

Kuviossa 14 olen käyttänyt kaikkia Canvas Scalerin -asetuksia edellisen luvun käyttöliittymän yläosaan. Kuvasta voi nähdä Match Width or Height -asetuksen eri säätöjen vaikutukset sekä vertailun Expand- ja Shrink-asetuksiin. Sisällytin Shrink-asetuksen vertailuun, vaikka en asetuksesta sen enempää tässä työssä kerro, koska se ei ole osoittautunut tarpeelliseksi asetukseksi mobiilipeliä suunnitellessa. Jacksonia (2015, luku 2, kappale ”Scale with Screen Size”) lainatakseni, se on hieman outo asetus. Uskon, että sille olisi käyttöä, jos käyttöliittymä tehtäisiin pienimmän mahdollisen natiiviresoluution perusteella, ja käyttöliittymän tarvitsisi skaalata aina vain suurempaan resoluutioon. Mobiililaitteiden kehitys kuitenkin on sen verran rivakkaa, että käsitys pienimmästä resoluutiosta voi vanheta puolessa vuodessa. Näin kävi myös The Walking Dead: Our World -pelin kehityksen aikana, kun iPhone X tuli markkinoille. Shrink-asetus on kuitenkin hyvä pitää mielessä.

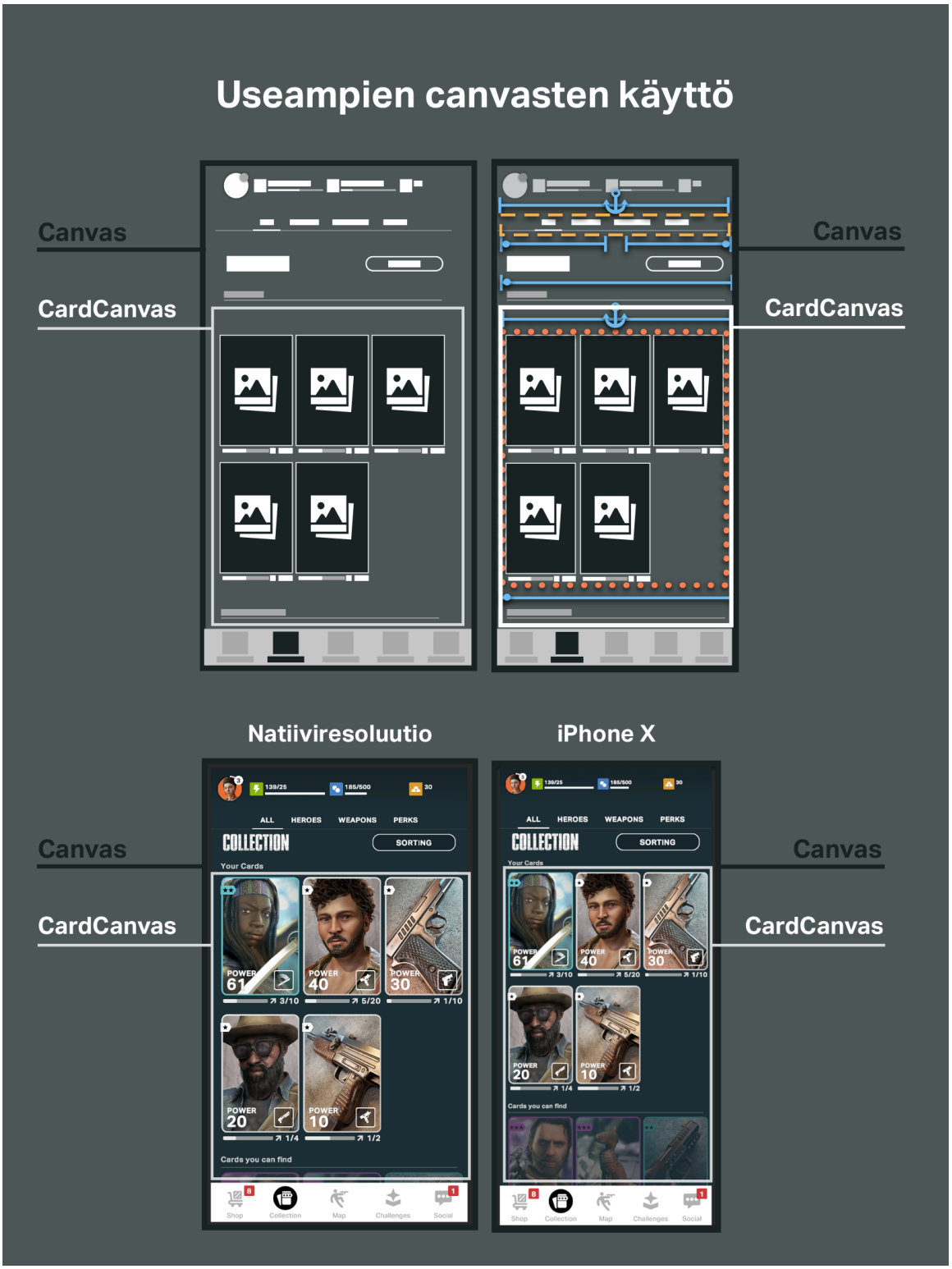
Kuvio 14. Canvas Scaler -asetusten erot käyttöliittymän yläosassa

Huomioitavaa on myös, että Expand-asetus saa käyttöliittymän näyttämään samalta kuin jos Match Width or Height -asetuksen säädin on vedetty leveyden ääripäähän eli nollaan. Vastaavasti Shrink -asetus näyttää taas vastaavan Match Width or Height asetuksen korkeuden ääripäätä eli ykköstä. Näin ollen koen, että Match Width or Height -asetus on oleellisin silloin, kun ääripäät eivät riitä vaan skaalausta täytyy pystyä tasapainottamaan korkeuden ja leveyden välillä.

5.1.3 Valittu Canvas Scaler -asetus ei sovi kaikille elementeille

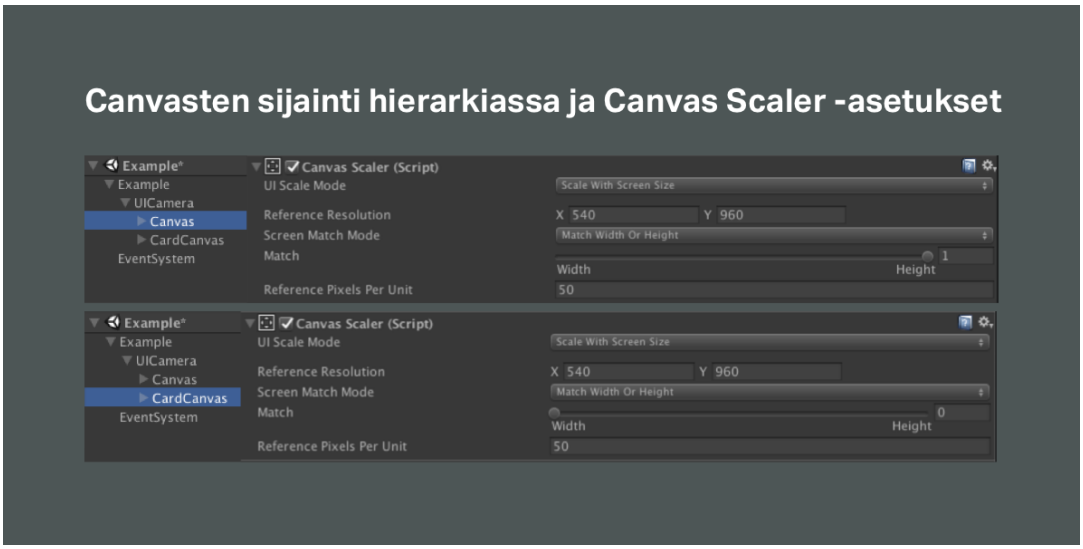
Toisinaan käyttöliittymää tehdessä tulee tilanteita, joissa Canvas Scaler -asetuksista yksi sopisi paremmin yhdelle elementille ja toinen asetus taas toiselle. Kumpikaan asetus ei siis yksinään tuota toivottua skaalausta, mutta yhdessä asetukset saisivat aikaan halutun lopputuloksen.

Canvas-komponentteja on mahdollista luoda useampia samaan käyttöliittymään, joten halutessaan suunnittelija voi asettaa yhden Canvas Scaler -komponentin Expand-asetukselle ja toisen Match Width or Height -asetukselle. Edellisten lukujen esimerkeissä ollut käyttöliittymä on tehty kahdella Canvaksella. Kaksi Canvasta mahdollistaa sen, että kortit saadaan täyttämään koko näytön leveys, mutta muiden elementtien yhteneväisyys natiiviresoluution kanssa onnistuu parhaiten suosimalla näytön korkeutta. Kuviossa 15 on nähtävillä kahden Canvaksen sijainti käyttöliittymässä.



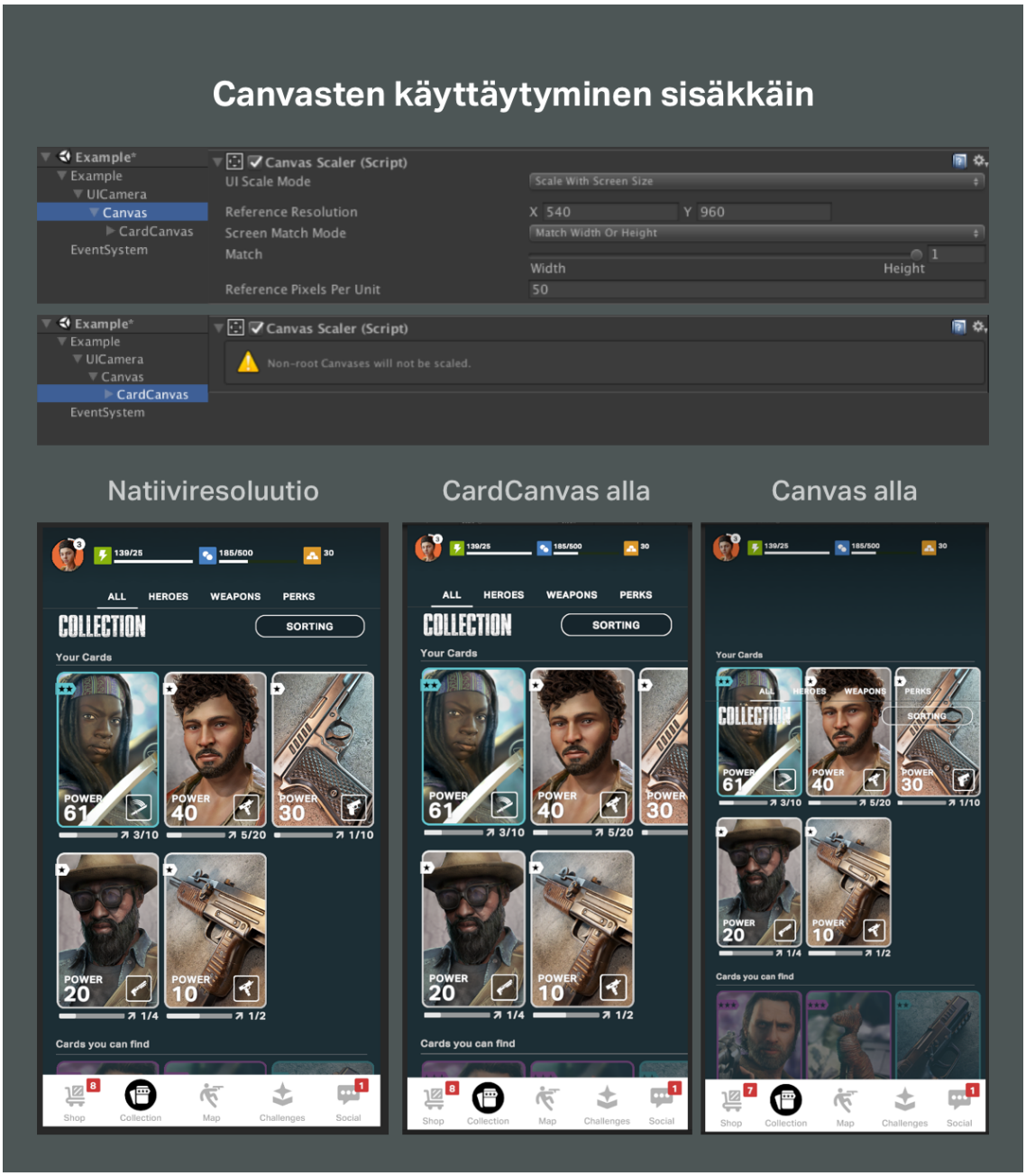
Kuvio 15. Kahden Canvaksen sijainti hierarkiassa

Molemmissa Canvas Scaler -komponenteissa on asetuksena Match Width or Height -asetus, mutta toinen on asetettu suosimaan korkeutta, ja korttien listausta varten tehty Canvas suosii leveyttä. Kuviossa 16 on nähtävillä molempien asetukset.



Kuvio 16. Canvasten sijainti hierarkiassa sekä Canvas Scaler -asetukset

Periaatteessa on mahdollista tehdä myös sisäkkäisiä Canvaksia, jotka määräävät toistensa skaalautumisesta (Jackson 2015, luku 2, kappale ”The Canvas”.) Mutta silloin, kun halutaan käyttää eri Canvas scaler -asetuksia samassa käyttöliittymässä, Canvasten täytyy olla rinnakkain hierarkiassa. Jos Canvakset ovat sisäkkäin, ylin määrää skaalautumisen, ja alapuolella oleviin ei voi asettaa omia skaalautumisen asetuksia. Tästä esimerkki kuviossa 17.



Kuvio 17. Vain ylin sisäkkäisistä Canvaksista määrää skaalautumisen, ja alemman Canvas Scaler -asetukset ovat poissa käytöstä

Kuviosta 17 voi myös nähdä miltä käyttöliittymä näyttäisi, jos käytössä olisi vain yksi Canvas, koska vain yhden Canvaksen skaalautumisen asetukset ovat voimassa käyttöliittymässä.

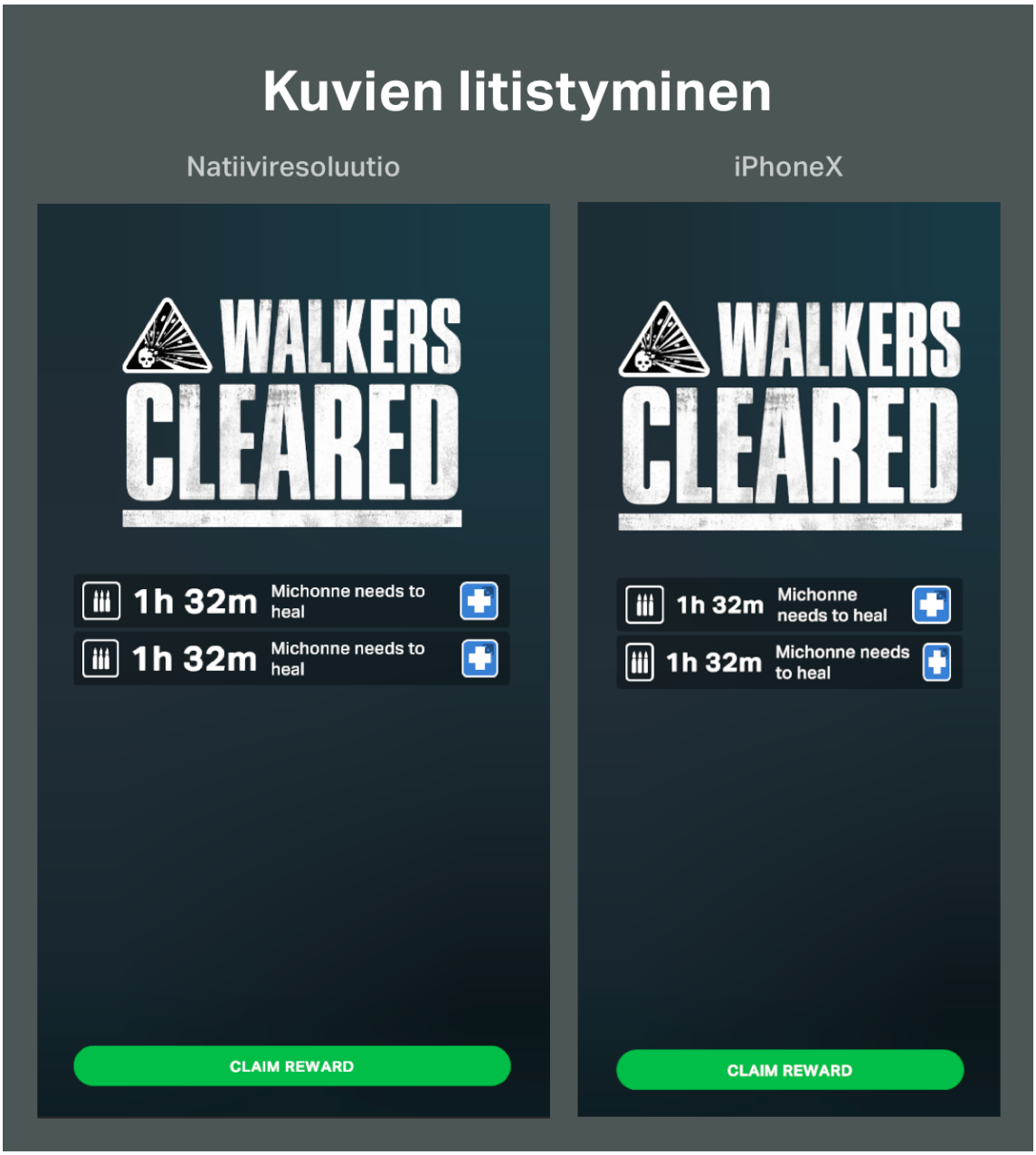
5.2 Elementtien skaalautumisen ongelmat suhteessa toisiinsa

Tämä luku käsittelee niitä ongelmia, jotka syntyvät silloin, kun Automatic Layout Groups -järjestelmän alla olevat elementit eivät skaalaudu odotetulla tavalla. Tämän teeman ongelmia tuli vastaan neljä kertaa havainnoinnin aikana. Tosin jotkut yksittäisten elementtien ongelmista tuli myös ratkaistua tämän teeman alla, mutta teemoittelu on tehty ongelmasta eikä ratkaisusta lähtöisin, joten laskin teeman alle vain ongelmien esiintymiskerrat.

5.2.1 Elementeistä osan pitäisi skaalata ja osan ei

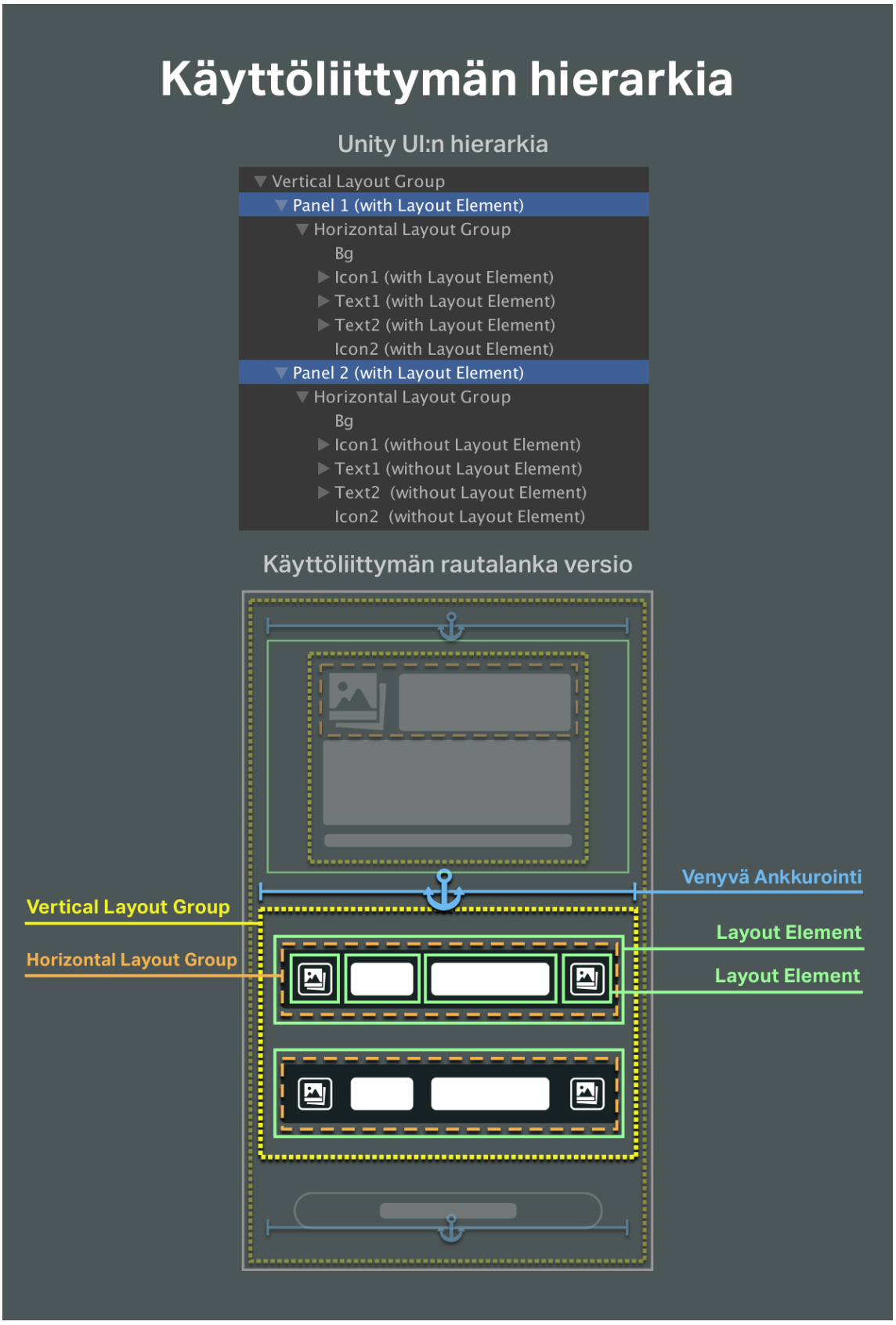
Tämä ongelma juontaa juurensa siitä, että Automatic Layout Group -komponentin sisällä olevat elementit eivät automaattisesti tiedä, milloin niiden kuuluu tai ei kuulu skaalautua.

Havainnointini aikana tästä nousi esiin hyvä esimerkki. Horizontal Layout Group -komponentin sisällä oli kaksi kuvaa ja kaksi tekstielementtiä. Skaalatessa kapeampaan resoluutioon kuvat litistyivät, koska kuvat antoivat tilaa teksteille, jotka taas venyivät liikaa kuvien välissä. Tarkoitus oli saada kuvat pitämään haluttu koko kaikissa resoluutioissa, eli ne eivät saaneet skaalautua lainkaan, ja tekstien piti venyä kuvien välissä sen verran mitä kuvien väliin jäi tilaa. Kuviossa 18 nähtävillä esimerkki kuvien litistymisestä.



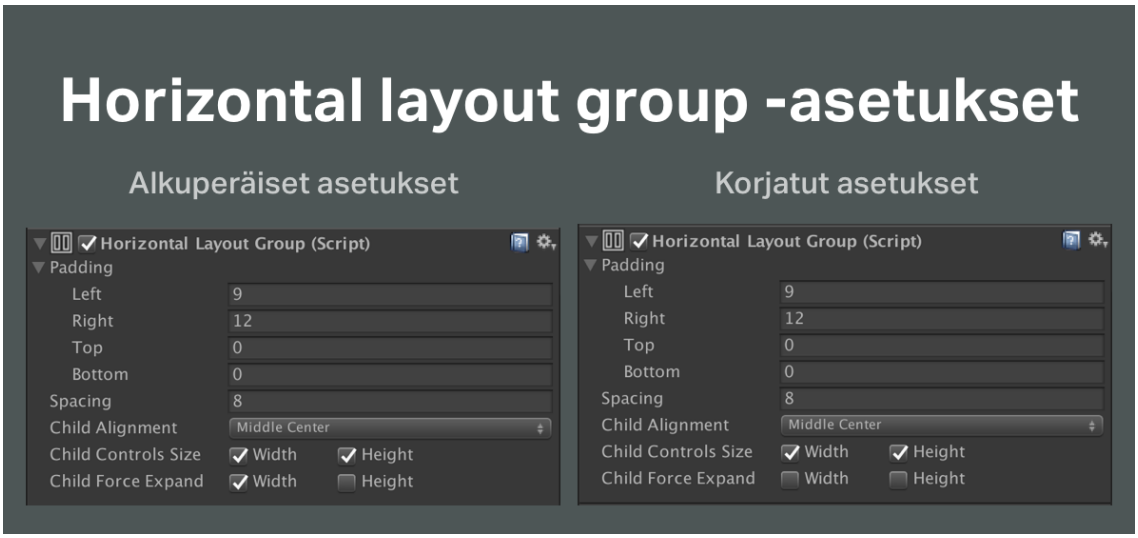
Kuvio 18. Natiiviresoluutio ja iPhone X:lle skaalattu käyttöliittymä, missä ensimmäisessä elementissä on nähtävissä onnistunut skaalaus, ja alemmassa ongelman lähtötilanne. Kuvasta voi nähdä miten alemman elementin kuvat ovat litistyneet.

Kuviossa 19 on käyttöliittymän hierarkia sekä rautalankaversio käyttöliittymästä helpottamaan seuraavien vaiheiden seuraamista.



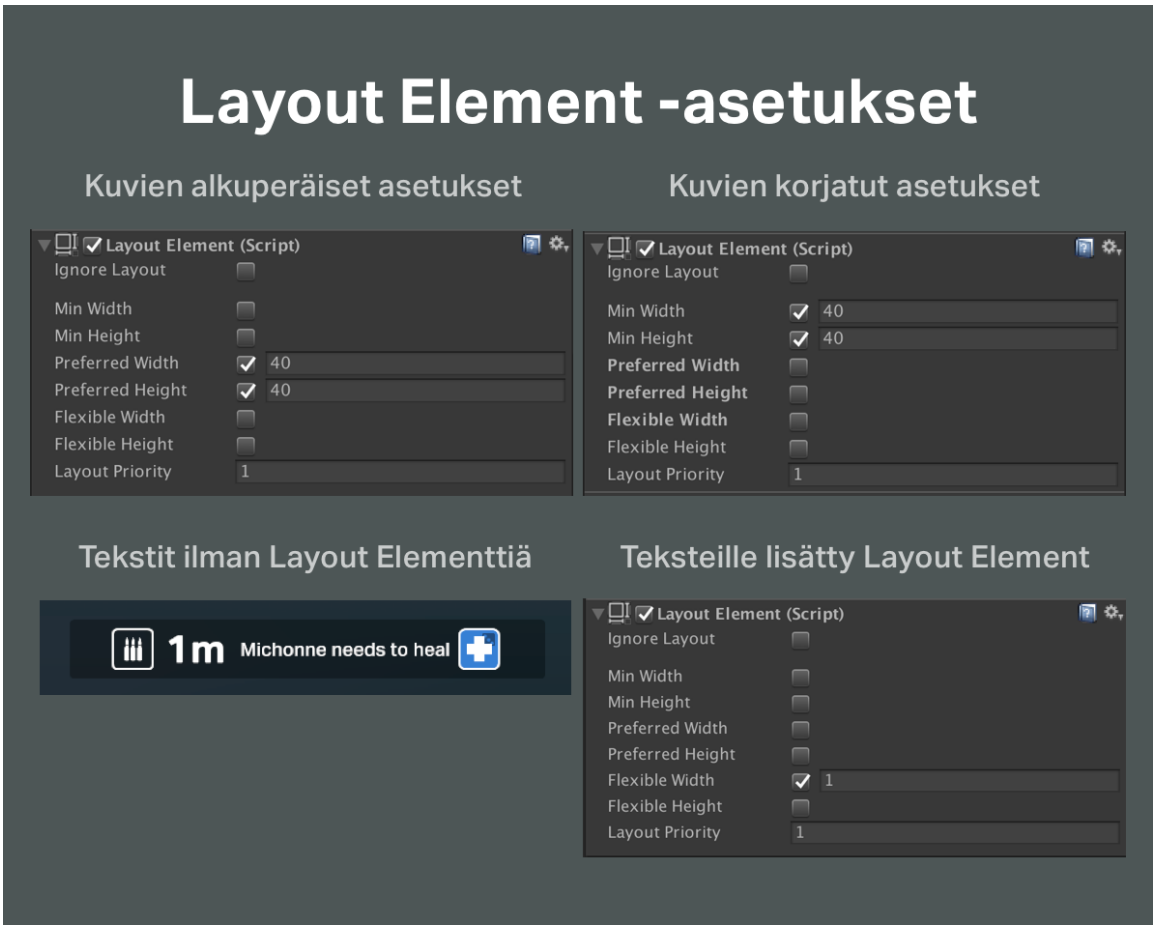
Kuvio 19. Hierarkia ja rautalankaversio

Ensin Horizontal Layout Group -komponentin asetuksiin tehtiin sellainen muutos, että Child Controls Size -asetus vaikuttaa sekä leveyteen että korkeuteen, mutta Child Force Expand -asetus jätetään kokonaan tyhjäksi. Child Controls Size -asetukset yhdistettynä layout element -komponenteilla tehtäviin määrittelyihin, saadaan elementit määräämään oman kokonsa. Tämä on tärkeää siksi, että Automatic Layout Group -komponentin asetukset voivat yliajaa Layout Element -komponentin määritysten yli. Esimerkiksi tässä tapauksessa kuville asetetut Layout Element -komponentit eivät skaalatessa riittäneet pitämään kuvia halutussa koossa, jos Child Force Expand -asetus on päällä. Kuviossa 20 esillä asetusten muutos.



Kuvio 20. Horizontal layout group -asetukset

Kuvien Layout Element -komponenttiin vaihdettiin Min Size -asetus, joka määrää kuville minimileveyden ja -korkeuden. Sen avulla kuvat skaalaavat niin kauan, kunnes niiden Min Size -asetuksen koko tulee vastaan. Tällä tavalla kuvat säilyttävät kokonsa, toisin kuin Preferred Size -asetuksella, joka ei skaalatessa pidä kuvien haluttua kokoa, vaikka näin nimestä voisikin ajatella. Kuvien ja tekstien asetuksien muutokset ovat nähtävillä kuviossa 21.



Kuvio 21. Layout element -asetukset

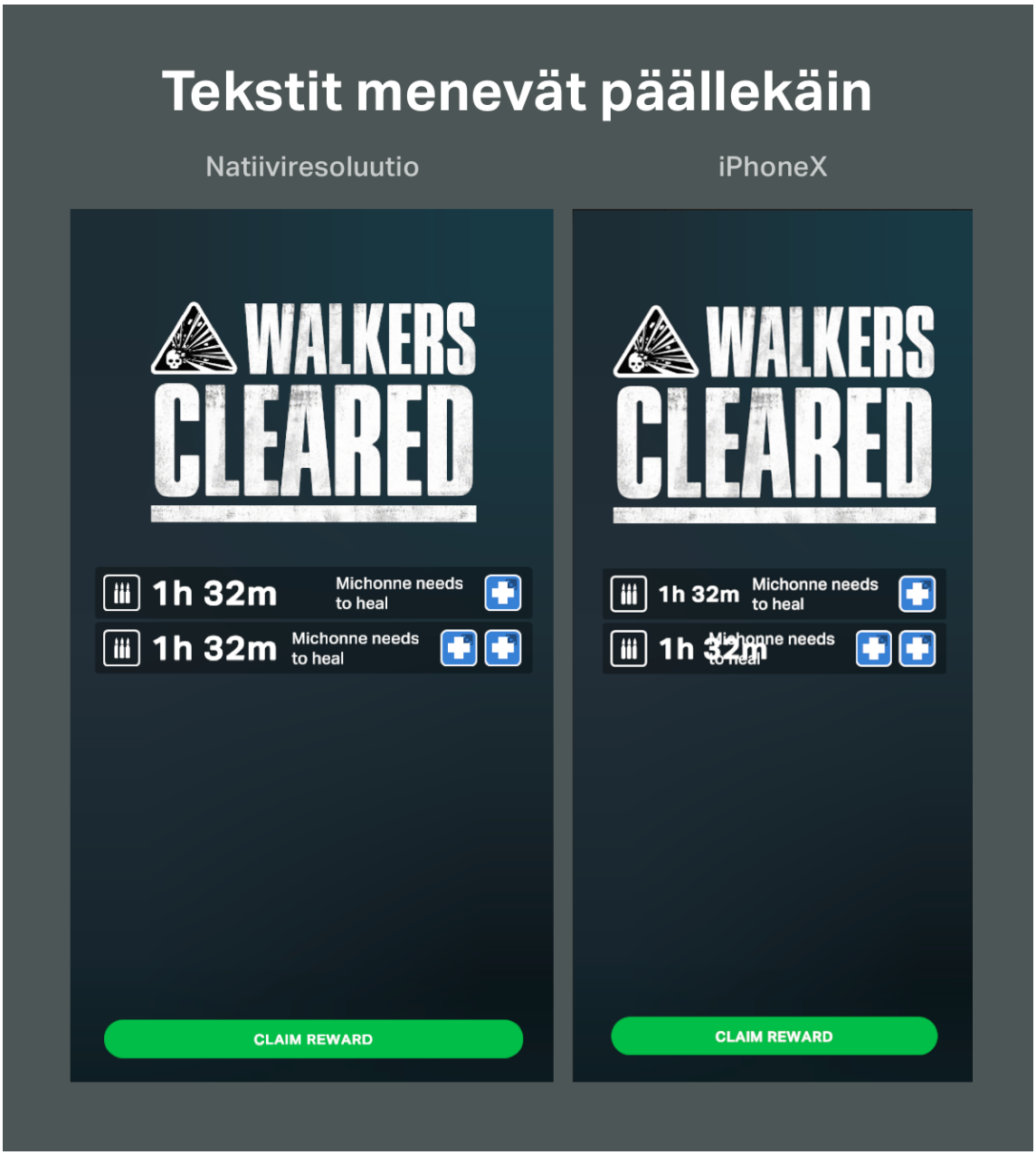
Preferred size -asetus ei myöskään ole tarpeellinen tässä tapauksessa, koska tekstielementteihin asetettiin Layout Element -komponentit, joissa on Flexible Width -asetuksessa 1. Tällöin tekstielementit venyvät jäljelle jäävään tilaan, eikä kuvien siten ole mahdollista ottaa minimikokoaan suurempaa tilaa, joten Preferred Size -asetus ei pääsisi edes vaikuttamaan kuviin. Tekstien flexible width -asetus on tärkeä siksi, että jos tekstin määrä vähenisi, elementit eivät alkaisi siirtyä paneelin keskelle. Tämä tapahtuisi siksi, koska Horizontal Layout Group -komponentissa ei ole Child Force Expand -asetusta, joka pakottaisi elementit käyttämään koko Automatic Layout Group -komponentin sisältävän elementin leveyttä.

Loppupäätelmä tästä on se, että Layout Element -komponentin Flexible Width -asetuksella voi saada yksittäisillä elementeillä aikaan saman skaalautumisen kuin Child Force Expand -asetuksella. Tällä tavalla on mahdollista hallita paremmin sitä, mitä elementtejä skaalataan ja mitä ei.

5.2.2 Elementit menevät toistensa päälle tai katoavat näkyvistä

Lähtökohtaisesti Automatic Layout Group -komponentti huomioi sisällään olevien elementtien koot, mutta skaalatessa tila saattaa yksinkertaisesti loppua kesken. Silloin Automatic Layout Group -komponentti pyrkii ensisijaisesti siihen, että elementit pysyvät sen rajojen sisällä, mutta elementit saattavat silloin tulla pusketuksi toistensa päälle. Jos osa elementeistä ei jousta ja osa joustaa, on mahdollista, että kokonsa pitävät elementit puristavat joustavat elementit jopa näkymättömiin.

Otetaan edellisen luvun käyttöliittymä esimerkiksi tilanteesta, jossa kaksi kokonsa pitävää kuvaa ja kaksi venyvää tekstielementtiä ovat saman Automatic Layout Group -komponentin sisällä, ja lisätään listaan vielä yksi kuva perään.



Kuvio 22. Ero elementeissä, kun toiseen riviin on lisätty yksi kuva lisää

Edellä olevassa kuviossa 22 voi nähdä, miten tekstit menevät toistensa päälle. Tämä johtuu yksinkertaisesti siitä, että tässä käyttöliittymässä tilan raja tulee vastaan tällä hierarkialla. Kuvasta näkee myös, että natiiviresoluutiolla ongelmaa ei ole, joten jos käyttöliittymää ei skaalata, tätä ongelmaa ei välttämättä huomata.

Tätä ongelmaa pyrin ratkaisemaan muuttamalla kahden viimeisen kuvan Layout Element -asetuksia. Otin Min Size -asetukset pois ja tilalle laitoin Flexible Width -asetuksen arvolla 0. Tällä tavalla kuvat joustavat tekstien ohella, ja tila jaetaan tasan joustavien

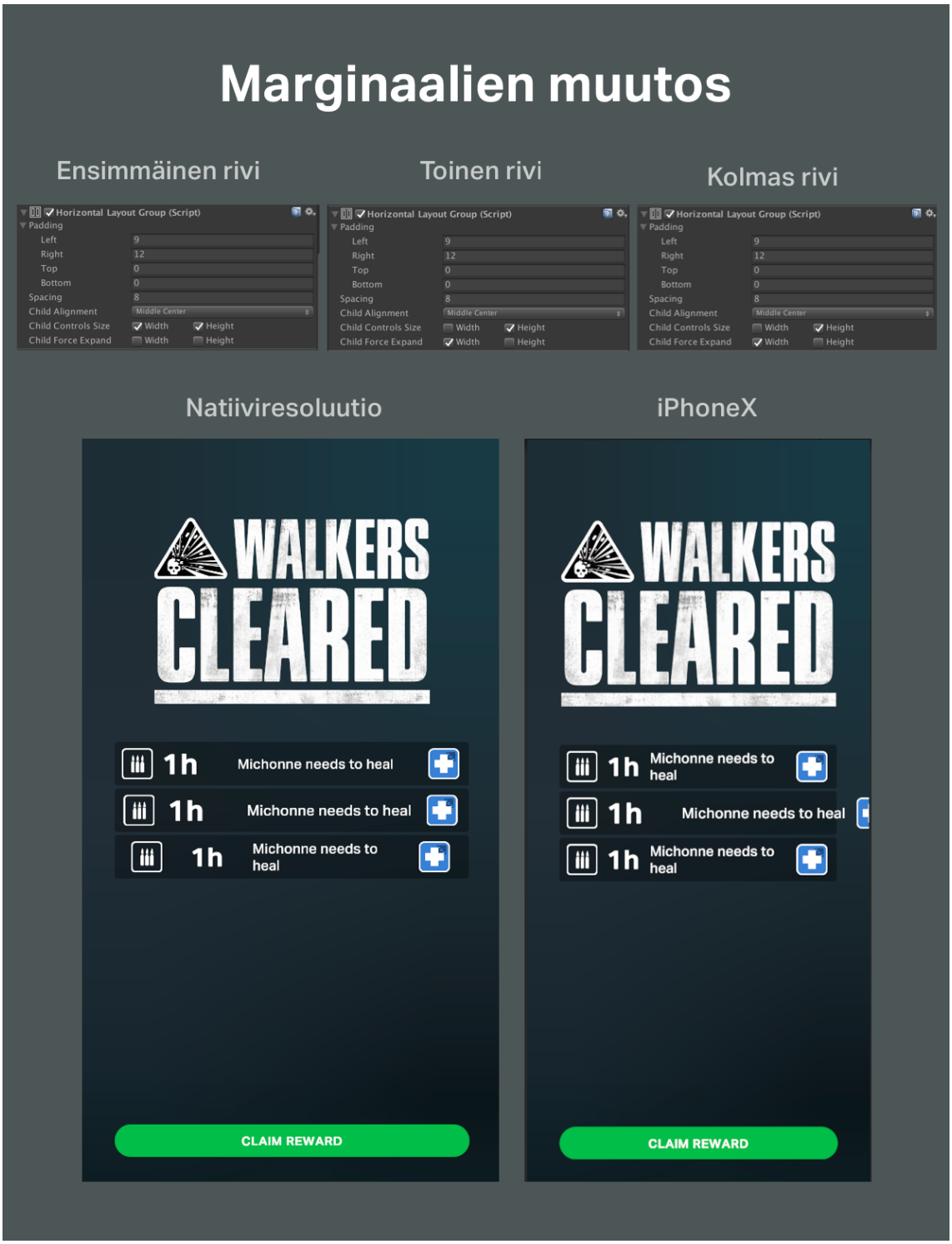
elementtien välillä, jolloin elementit mukautuvat paremmin vierekkäin. Kuviossa 23 asetusten muutos.

Kuvio 23. Layout Element -asetuksen muutos

Jos tulee tilanne, että elementtejä on liikaa ja skaalaaminen kapeammaksi ei onnistu edellä kuvatusta ratkaisusta huolimatta lainkaan, suosittelen kokeilemaan Canvas Scaler -komponentin Expand-asetusta. Edellä kuvatussa esimerkissä skaalaus hoidetaan Match Width or Height -asetuksella suosien korkeutta.

5.2.3 Elementtien väliset marginaalit ovat liian isoja

Godbold (2018, 84) selittää kirjassaan, että Automatic Layout Group -komponentin Spacing-asetuksella voidaan määrittää elementeille välimatka eli marginaali. Tämä asetus voi tulla yliajetuksi, jos Child Force Expand -asetus on käytössä ilman Control Child Size -asetusta. Elementtien marginaalit voivat tällöin kasvaa suuremmiksi kuin haluttaisiin. Kun Control Child Size on myös käytössä, marginaalit käyttäytyvät odotetusti. (Godbold, 2018, 86.) Kuviossa 24 on tästä esimerkki, jossa näkyy edellisten lukujen käyttöliittymässä eri Automatic Layout Group -asetusten vaikutus marginaaleihin. Huomioitavaa on, että toisella ja kolmannella rivillä on samat asetukset, mutta toisen rivin asetusten muutos on tehty, kun natiiviresoluutio on ollut käytössä. Kolmannen rivin muutokset taas on tehty, kun käyttöliittymä on ollut skaalattuna iPhone X:lle. Toisin sanoen toinen rivi on tehty natiiviresoluution mittojen mukaan ja toinen iPhone X:n.



Kuvio 24. Horizontal Layout Group -komponentin asetusten vaikutukset elementtien välisiin marginaaleihin

Kuviosta 24 näkee, etteivät ensimmäisenkään rivin marginaalit pysy samoina skaalatessa. Ensimmäinen tekstielementti on huomattavasti lähempänä toista kuin mitä natii-

viresoluutiossa. Toisinaan marginaalien suhteen täytyy tehdä kompromisseja, kun skaalataan kapeammille resoluutioille. Natiiviresoluutioon voi joutua tekemään levemmät marginaalit kuin ehkä haluaisi. Tähän voisi jälleen olla ratkaisuna Expand -asetuksen käyttö Canvas Scaler -komponentissa, mutta suoriltaan se ei tähän käyttöliittymään sovi, koska käyttöliittymä on painotukseltaan pitkänmallinen.

Kuvio 25. Canvas Scaler -komponentti vaihdettu Expand-asetukselle

Kuviossa 25 on esimerkki Expand -asetuksella tehdystä skaalauksesta samassa käyttöliittymässä. Huomioitavaa on välimatka otsikon ja listan välillä, joka on kasvanut tarpeettoman suureksi skaalauksen myötä. Tällaisessa tilanteessa suunnittelijan täytyy tehdä

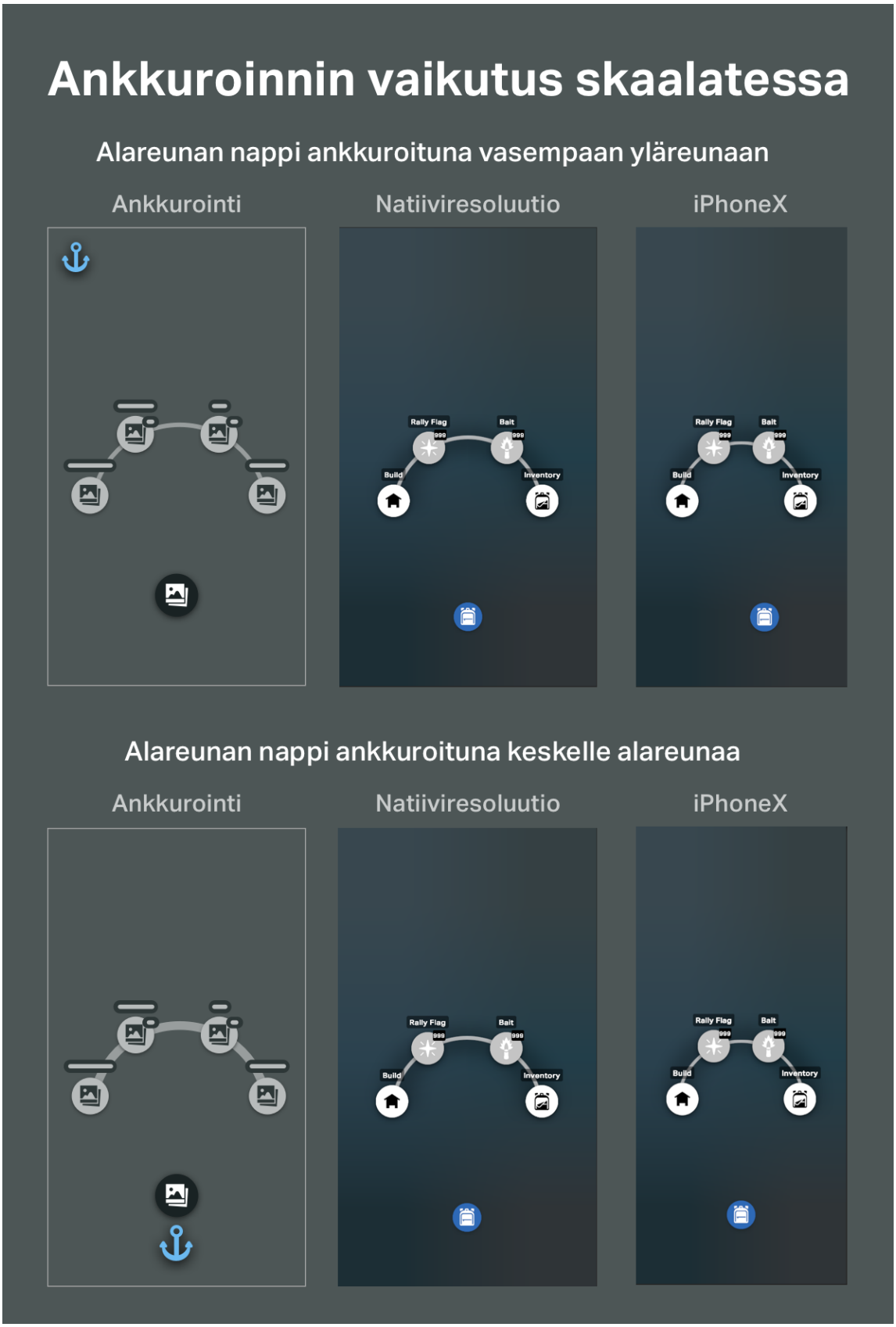
valinta sen väliltä tekeekö mieluummin kompromisseja elementtien marginaalien etäisyyksissä, vai rakentaako käyttöliittymän kokonaisuutena Expand -asetukselle sopivaksi. Voisi myös kokeilla käyttää kahta Canvasta, joista toinen hallitsee otsikkoa ja toinen listaa.

5.3 Yksittäisten elementtien skaalautumisen ongelmat

Tämän teeman ongelmia voisi ajatella niin, että näitä ongelmia tulee vastaan riippumatta siitä, onko kyse yksittäisestä kuvasta vai ylemmän tason Automatic Layout Groups -järjestelmästä. Pääasia on, että ongelma on vain tässä yhdessä elementissä - ei koko käyttöliittymässä, eikä elementtien välisissä suhteissa. Tämä ei toki tarkoita, etteikö yksittäisten elementtien ongelmia voisi ratkaista Automatic Layout Groups -järjestelmällä tai Canvas -komponentilla. Teeman ongelmia esiintyi 11 kertaa havainnoinnin aikana.

5.3.1 Elementin sijainti muuttuu skaalatessa epäjohdonmukaisesti

Jos elementti on osa Automatic Layout Groups -järjestelmää, suosittelen katsomaan elementtien skaalautumisen ongelmat suhteessa toisiinsa luvusta ratkaisuja. Jos elementti ei ole osa Automatic Layout Groups -järjestelmää, ratkaisu ongelmaan löytyy ankkuroinnista. Elementit voidaan vain raahata halutuille paikoilleen, mutta skaalatessa elementtien ankkuroinnilla on merkitystä, koska ankkurointi vetää elementtiä puoleensa. Siksi skaalatessa elementti, joka on ankkuroitu vasempaan yläkulmaan, liikkuu vasemman yläkulman mukaan, vaikka elementti olisi raahattu oikeaan alakulmaan. Kuvio 26 havainnollistaa, mitä ankkurointi tekee elementeille skaalatessa.



Kuvio 26. Ankkurointi muuttaa napin paikkaa näytöllä skaalatessa

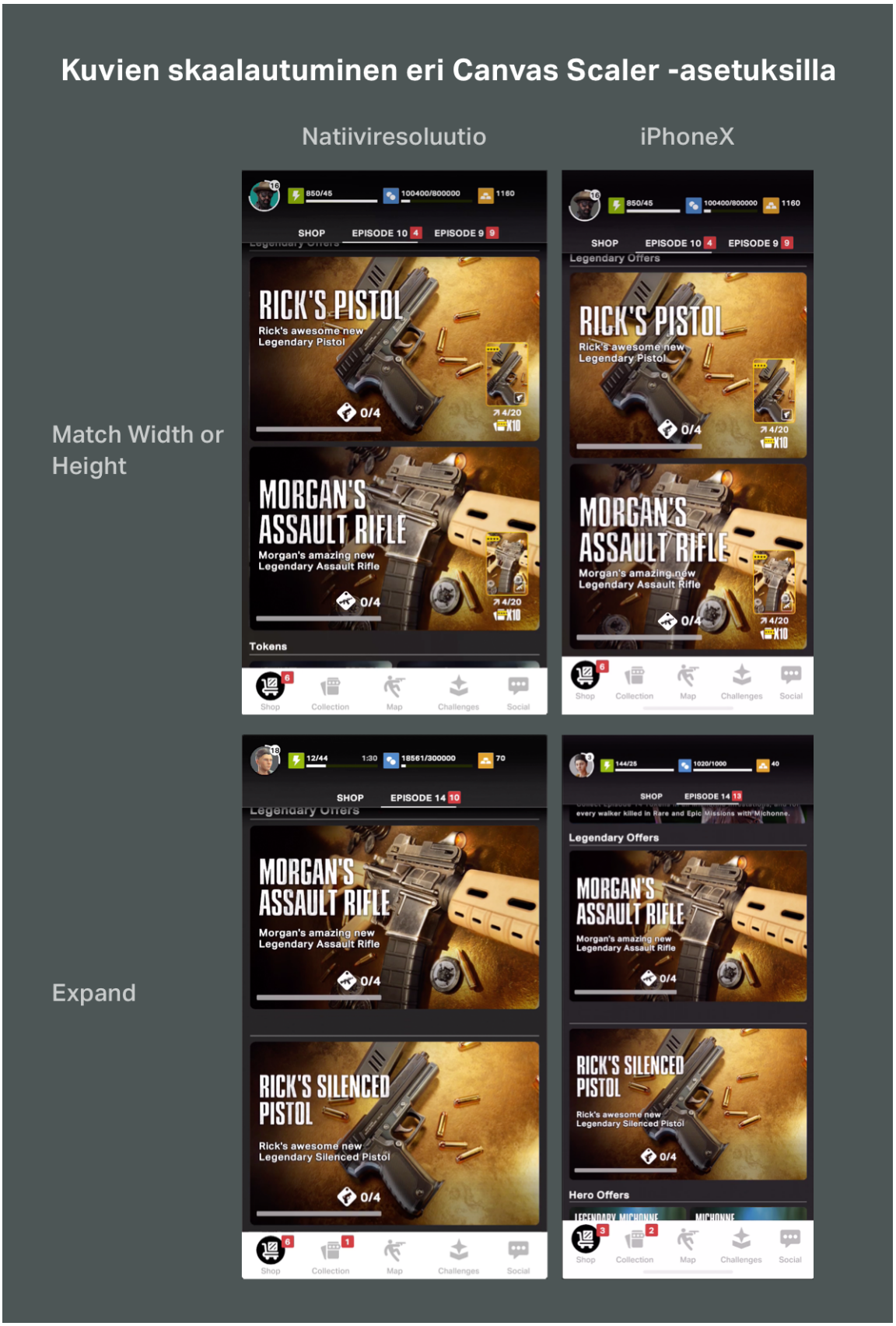
Kuviosta 26 näkee, että vasempaan yläreunaan ankkuroitu nappi siirtyy lähemmäksi oikeaa reunaa skaalatessa. Ankkurointi pyrkii säilyttämään saman etäisyyden ankkuripisteeseen skaalatessa, joten nappi siirtyy siksi kohti oikeaa reunaa. Vastaavasti taas, kun nappi on ankkuroitu keskelle alareunaan, nappi pysyy paikoillaan skaalatessa. Ankkurointi pitää napin paikoillaan, koska napin etäisyys ankkuripisteeseen ei muutu skaalatessa. Erikoisilta siirtymiltä voi välttyä ankkuroimalla elementit aina lähimpään kulmaan tai reunaan. Kiinteästi ankkuroitu elementti seuraa sitten tätä yhtä pistettä, ja venyväksi ankkuroitu elementti siirtyy näytön reunojen välillä joko pituus- tai leveyssuunnassa tai kattamalla koko näytön.

5.3.2 Elementti rajautuu maskin sisällä skaalatessa

Havainnoinnin aikana käsittelin tätä ongelmaa varsin paljon. Tehtävänä oli pyrkiä ratkaisemaan, miten maskin sisällä oleva kuva saataisiin pitämään kokonsa skaalatessa ilman, että kuva rajautuu maskin reunasta kapeampaan resoluutioon siirryttäessä. Rajautuminen tapahtui siksi, että kuva tehtiin koossa, joka sopi natiiviresoluutiolle, mutta kapeampaan resoluutioon siirryttäessä skaalautuva maski rajasi skaalautumattoman kuvan uudelleen. Kuviossa 27 tarkempi kuvaus ongelmasta.

Kuvio 27. Vihreällä esitetty skaalautumaton kuva rajautuu skaalautuvan maskin sisällä

Tähän ongelmaan oli monia mahdollisia ratkaisuja, mutta eri vaihtoehdoista lopulta päädyttiin Canvas Scaler -komponentin Expand -asetukseen siirtymiseen. Näin jälkikäteen on helppo sanoa, että tämä päätös oli hyvä ratkaisu, sillä Expand-asetus korjasi ongelman varsin mutkattomasti. Käyttöliittymään ei tarvinnut itse skaalauksen metodin vaihtamisen lisäksi tehdä juuri muuta kuin säätää hieman marginaaleja. Tämän helppouden vuoksi koen, että jos käyttöliittymään tehdään elementtejä, jotka eivät syystä tai toisesta saa skaalautua, niiden koko täytyy määritellä pienimmän mahdollisen resoluution mukaan tai käyttää Expand -asetusta, jotta elementit varmasti piirtyvät kokonaisina näytölle. Kuviosta 28 voi nähdä miten Expand-asetus muutti käyttöliittymän skaalausta.



Kuvio 28. Match Width or Height - ja Expand-asetukset vertailussa

Kuviosta 28 näkee miten tekstien sijainti kuvien päällä on yhtenäistynyt Expand-asetuksella. Match Width or Height -asetuksella tekstit tavallaan vuotavat enemmän kuvien päälle, koska tekstin alla olevan kuvan vasemmalta puolelta on rajautunut kuva-alaa pois. Expand-asetuksella tehdyssä käyttöliittymässä rajausta ei tapahdu lainkaan, joten tekstit vievät kuvien päältä saman verran tilaa. Huomioitavaa on se, että Expand-asetuksen vuoksi tekstit ovat pienentyneet jonkin verran, mikä on syytä huomioida pistekoon kanssa.

6 Johtopäätökset

Tarkoitukseni oli tällä työllä pyrkiä vähentämään testauksen tarvetta taulukoimalla eri skaalauksen ongelmia sekä niiden mahdollisia ratkaisuja niin, että ongelman ilmetessä taulukosta voisi löytää nopeasti ratkaisun. Tässä luvussa esittelen ja käyn lyhyesti läpi analyysissa esitettyjen ongelmien teemojen mukaan tehdyt taulukot, jotka tiivistävät analyysissa laajemmin läpikäydyt käyttöliittymän skaalautumisen ongelmat sekä niihin löydetyt ratkaisut.

Taulukot eivät sisällä kaikkea analyysin sisältöä, vaan ne yksinkertaistavat ratkaisuvaiheet helposti silmäiltävään muotoon, jotta taulukkoja voidaan käyttää nopeaan ongelmanratkaisuun. Taulukot kuitenkin sisältävät ongelmat ja ratkaisuvaiheet siinä järjestyksessä, jossa ne on esitetty analyysissä. Näin ollen jos jokin taulukon kohta tarvitsee selvennystä, analyysista voi hakea sekä tarkennusta että syy- ja seuraussuhteita ongelmien ja ratkaisujen välillä.

Taulukko 1. Koko käyttöliittymän skaalautumisen ongelmat ja ratkaisuvaiheet

#1 Koko käyttöliittymän skaalautumisen ongelmat		
Ongelman kuvaus	**Vaihe 1**	**Vaihe 2**
Käyttöliittymä skaalaa ennalta arvaamattomasti	Tarkista, että Canvaksen Canvas Scaler -komponentin UI Scale Mode -asetus on asetettu Scale With Screen Size -tilaan.	Aseta tarvittaessa elementteihin venyvä ankkurointi.
Käyttöliittymän elementit skaalautuvat tarpeettoman suuriksi tai pieniksi	Jos Canvas Scaler -komponentissa on Match Width or Height -asetus, joka saa elementit näyttämään skaalatessa suuremmilta kapeammalla resoluutiolla, vaihda Expand-asetukselle.	Jos Canvas Scaler -komponentissa on Expand-asetus, vaihda Match Width or Height -asetukselle ja tarpeen tullen tasapainota skaalausta.
Valittu Canvas Scaler -asetus ei sovi kaikille elementeille	Luo hierarkiaan rinnakkain useampi Canvas ja aseta niille eri Canvas Scaler -asetukset.	Jaa elementit Canvasten sisään sen mukaan, mikä Canvas Scaler -asetus skaalaa elementit mahdollisimman yhteneväisesti natiiviresoluution kanssa.

Taulukot on tehty sillä oletuksella, että Unity UI:n työkalut ovat tuttuja, ja tarkoitus on saada ratkaisu nopeasti mahdolliseen ongelmaan. Ajatus on antaa ongelma ja ratkaisu sellaisessa muodossa, jossa antaisin ne kollegalle, jos hän kysyisi neuvoa tulenpalavassa kiireessä.

Taulukko 2. Elementtien skaalautumisen ongelmat suhteessa toisiinsa ja ratkaisuvaiheet

#2 Elementtien skaalautumisen ongelmat suhteessa toisiinsa			
Ongelman kuvaus	**Vaihe 1**	**Vaihe 2**	**Vaihe 3**
Automatic Layout Group -komponentin sisällä olevista elementeistä vain osan halutaan skaalaavan ja osan pitävän tietyn koon	Automatic Layout group -komponenttiin laitetaan Child controls size -asetus päälle sekä leveyteen että korkeuteen ja Child Force Expand -asetukset laitetaan kokonaan pois päältä	Luodaan skaalautumattomille elementeille Layout Element -komponentti ja asetetaan Min Size -asetuksella haluttu minimikoko.	Elementeille laitetaan Layout Element -komponenttiin Flexible Width -asetus arvolla 0 silloin, kun ei haluta, että elementti venyy ja arvolla 1, kun halutaan, että elementti venyy.
Automatic Layout Group -komponentin sisällä olevat elementit menevät toistensa päälle tai katoavat näkyvistä	Tehdään kokonsa pitävistä elementeistä skaalattavat. Layout Element -komponetteihin Min size -asetuksien tilalle Flexible Width -asetus arvolla 0.	Vaihdetaan Canvas Scaler -komponenttiin Expand-asetus, jos käytössä on Match Width or Height.	Katso taulukosta 1 kohta "Valittu Canvas Scaler -asetus ei sovi kaikille elementeille", jos Canvas Scaler -asetuksen vaihto aiheuttaa uusia ongelmia.
Elementtien väliset marginaalit ovat liian isoja	Automatic Layout Group -komponentin Child Force Expand -asetus ja sitä vastaava Control Child Size -asetus täytyy olla yhtä aikaa päällä, jotta Spacing-asetuksella määritelty marginaalit käyttäytyvät odotetusti.	Jos vaihe 1 ei riitä korjaamaan ongelmaa, vaihdetaan Canvas Scaler -komponenttiin Expand-asetus, jos käytössä on Match Width or Height.	Katso taulukosta 1 kohta "Valittu Canvas Scaler -asetus ei sovi kaikille elementeille", jos Canvas Scaler -asetuksen vaihto aiheuttaa uusia ongelmia.

Tämän teeman taulukossa ongelmille lisättiin vaihe 3, koska ensimmäinen ongelma vaatii useamman muutoksen kuin muut ongelmat, ja kahteen muuhun taas voi saada suunnittelijaa enemmän tyydyttävän lopputuloksen käyttämällä useampaa Canvasta, joten vaiheessa 3 kehotetaan siirtymään taulukkoon 1 katsomaan lisäohjeita.

Taulukko 3. Yksittäisten elementtien skaalautumisen ongelmat ja ratkaisuvaiheet

#3 Yksittäisten elementtien skaalautumisen ongelmat		
Ongelman kuvaus	**Vaihe 1**	**Vaihe 2**
Elementin sijainti muuttuu skaalatessa epäjohdonmukaisesti	Jos elementti ei ole osa Automatic Layout Groups -järjestelmää, varmista että elementti on ankkuroitu lähimpänä olevaan näytön kulmaan tai reunaan.	Jos elementti on osa Auto Layout -järjestelmää, katso "Elementtien skaalautumisen ongelmat suhteessa toisiinsa" taulukosta lisää ohjeita
Elementti rajautuu maskin sisällä skaalatessa	Vaihdetaan Canvas Scaler -komponenttiin Expand-asetus, jos käytössä on Match Width or Height.	Katso taulukosta 1 kohta "Valittu Canvas Scaler -asetus ei sovi kaikille elementeille", jos Canvas Scaler -asetuksen vaihto aiheuttaa uusia ongelmia.

Vaikka tämän teeman ongelmia esiintyi havainnoinnin aikana eniten, on taulukko kaikista lyhyin. Elementin rajautuminen maskin sisällä vaati niin paljon käsittelyä, että olisin saattanut saada pelkästään siitä opinnäytetyön aikaiseksi. Toisaalta, koska ratkaisu oli varsin yksinkertainen, olen tyytyväinen, että havainnoinnin aikana tuli myös paljon muita ongelmia esille.

7 Lopuksi

Työ saavutti tavoitteensa eli ongelmien ja ratkaisujen taulukoinnin, mutta taulukoiden testaaminen työn ohessa jäi valitettavasti tämän opinnäytetyön ulkopuolelle. Pyrkimykseni vähentää testauksen tarvetta jää siksi tässä työssä vain pyrkimykseksi, koska en voi todentaa, että näin todella tapahtuisi. Mutta ehkä palaan asiaan tulevaisuudessa, jos vaikka jatkankin opiskelua.

Havainnoinnin aikana ei noussut esille kaikkia mahdollisia skaalautumisen ongelmia. Jotkin ongelmat jäivät työn ulkopuolelle, koska niihin ei löytynyt ratkaisua havainnoinnin aikana tai niihin lähdettiin hakemaan ratkaisua muuttamalla käyttöliittymää. Koska tämä työ käsittelee vain skaalautumista ja Unitya, muilla keinoin ratkaistut ongelmat eivät enää kuuluneet työn piiriin, vaikka skaalauksen ongelmia olivatkin. Lisäksi yhden käyttöliittymän skaalauksen korjaaminen jäi pois työstä siksi, että käyttöliittymä on osa kokonaisuutta, joka on vasta tuotantovaiheessa, eikä siten voi olla esillä julkisesti.

Työ on tehty vain yhteen projektiin perustuen, ja se kannattaa huomioida lopputulosta tarkasteltaessa. Valitut tekemisen tavat ja hierarkian ratkaisut vaikuttavat tietysti käyttöliittymien skaalautumiseen, joten on mahdollista, ettei löytämiäni ratkaisuja voida käyttää suoraa muissa projekteissa. Tämän takia rajasin työn koskemaan vain Unity UI:n omia työkaluja, jotka ovat saatavilla kaikille Unitylla työskenteleville. Näin ollen on todennäköisempää, että havainnoinnin aikana löydetyt ratkaisut ovat hyvinkin uudelleen käytettävissä projektista riippumatta. Ainakin niin kauan kuin projekti nojaa lähtökohtaisesti Unity UI:hin sellaisenaan eikä koodilla luotuihin tai mukautettuihin työkaluihin.

Autoetnografian menetelmien käyttö sekä tekemisen kautta aineiston kerääminen oli valaiseva kokemus. Oman työni havainnointi järjestelmällisesti auttoi sekä havainnoinnin aikana että sen jälkeen. Havainnoinnin aikana tulin tietoisemmaksi omasta tekemisestäni, ja jälkikäteen vaiheiden seuraaminen on auttanut näkemään koko työn tekemisen kaaren, alkaen ongelman havaitsemisesta siihen, että ratkaisu poikii uutta tietoa, jota voi soveltaa ja käyttää uudelleen. Vaikka en ehtinytkään testaamaan valmiita taulukoita opinnäytetyöni aikana, palasin havainnoinnin aikana kerättyyn raakadataan aika ajoin työskennellessäni skaalautumisen kanssa. Suoraan sanoen en malta odottaa, että pääsen kokeilemaan, miten paljon aikaa taulukoiden käyttäminen säästää työaikaani. Tai oikeastaan en malta odottaa, että pääsen käyttämään sen säästetyn ajan käsitöihin.

Lähteet

AfterDawn 2018. Kuvasuhde. Tekniikkatermien sanakirja. <https://fin.after-dawn.com/sanasto/selitys.cfm/kuvasuhde> (luettu 30.9.2018).

AfterDawn 2018. Resoluutio. Tekniikkatermien sanakirja. <https://fin.after-dawn.com/sanasto/selitys.cfm/resoluutio> (luettu 30.9.2018).

Anttila, Pirkko 2006. Tutkiva Toiminta – Ilmaisu, Teos, Tekeminen. Hamina: Akatiimi Oy.

Chang, Heewon 2008. Autoethnography as Method. California: Left Coast Press, Inc.

ESOK-verkosto 2019. Käsitteet ja sanasto. Stivi-suositus. <http://www.esok.fi/stivisuositus/termit/kasitteet-ja-sanasto> (luettu 17.4.2019).

Fine, Richard 2015. Going deep with IMGUI and Editor Customization. Unity Blog. <https://blogs.unity3d.com/2015/12/22/going-deep-with-imgui-and-editor-customization> (luettu 25.10.2018).

Godbold, Ashley 2018. Mastering UI Development with Unity. Birmingham: Packt Publishing Ltd.

Hannula, Jani 2015. Homotetia eli venytyskuvaus geometrisissa konstruktioissa. Matematiikkalehti Solmu 3. <https://matematiikkalehtisolmu.fi/2015/3/homotetia.pdf> (17.4.2019).

Jackson, Simon 2015. Unity 3D UI Essentials (ePub-versio). Birmingham: Packt Publishing Ltd.

Johansen, Rune Skovbo 2014. Overview of the New UI System. Unity Blog. <https://blogs.unity3d.com/2014/05/28/overview-of-the-new-ui-system/> (luettu 25.10.2018).

Kielitoimisto 2018. Kielitoimiston sanakirja. <https://www.kielitoimistonsanakirja.fi/netmot.exe?motportal=80> (luettu 30.9.2018).

Sapio, Francesco 2015. Unity UI Cookbook. Birmingham: Packt Publishing Ltd.

StackExchange 2015. Responsive web design is based on screen resolution or screen size. Graphic Design Stack Exchange. <https://graphicdesign.stackexchange.com/questions/57442/responsive-web-design-is-based-on-screen-resolution-or-screen-size#comment95949_57442> (luettu 29.10.2018).

Unity Technologies 2018. Download Unity. Unity. <https://unity3d.com/get-unity/download> (luettu 25.10.2018).

Unity Technologies 2018. Unity User Manual (2018.2). Unity Documentation. <https://docs.unity3d.com/Manual/index.html> (luettu 16.8.2018).

Haastattelut

Reinikainen, Jani 2018. Senior UI Designer. Next Games Corporation. Haastattelu: 9.4.2019.