Markus Siitonen

Mobiiliohjelmointi

Päiväkirja

[1 Johdanto 1](#_Toc167953033)

[2 Oppimispäiväkirja 1](#_Toc167953034)

[3 Ajankäyttö 2](#_Toc167953035)

[4 AI työkalujen käyttö 2](#_Toc167953036)

[5 Viiteluettelo 2](#_Toc167953037)

[6 Liitteet 2](#_Toc167953038)

[6.1 Liite 1: Harjoitus 1 2](#_Toc167953039)

[6.1.1 Laitteen perustiedot 2](#_Toc167953040)

[6.1.2 Laitteen käyttöjärjestelmä 3](#_Toc167953041)

[6.1.3 Ohjelmointikielet ja tarvittavat työkalut 3](#_Toc167953042)

[6.1.4 Laitteen ominaisuudet 3](#_Toc167953043)

[6.1.5 Laitteen ominaisuudet 3](#_Toc167953044)

# Johdanto

Tähän dokumenttiin on kirjattu oppimispäiväkirjaa kurssin sisältöön ja tehtäviin liittyen.

# Oppimispäiväkirja

17.05.2024

Katsottu aloitusluento (Luento-1). Asensin Android Studion ja testasin että Virtual Device toimii ja, että saan oman testilaitteeni yhdistettyä Android Studioon. Virtual Devicen asennuksen kanssa oli hiukan ongelmia, mutta sain lopulta onnistumaan googlettamalla ongelman.

20.05.2024

**Harjoitus 1** (ks. liite 1)

**Harjoitus 3.** Tein harjoituksen ohjeistuksessa olleen Kotlin ohjeen mukaan (<https://developer.android.com/codelabs/basic-android-kotlin-compose-first-app#0>) Harjoitus onnistui hyvin, ei ongelmia. Kotlinin syntaksin opettelu varmasti vie jonkin aikaa.

21.05.2024

Luennot 2 ja 3. AkunTaskukirjat -projektin työstämistä luennon perusteella (Java -> Kotlin).

**23.05.2024**

Aloitin kurssin Programming Fundamentals in Kotlin (<https://www.coursera.org/learn/meta-programming-fundamentals-kotlin/>), koska halusin suorittaa kurssin Kotlinia käyttäen.

**24.05.2024**

Jatkoin kurssiaProgramming Fundamentals in Kotlin.

**26.05.2024**

Sain kurssin Programming Fundamentals in Kotlin valmiiksi.

**27.05.2024**

**Harjoitus 4.** Harjoitus onnistui hyvin. Ei suurempia ongelmia. Käytin apuna koodiesimerkkejä, sekä hiukan ChatGPT:n avustusta, kun selvittelin miten activity\_main.xml ja MainActivity.kt ’keskustelevat’ keskenään.

**28.05.2024**

**Harjoitus 5.** Aikaa kului turhaan Android Studion takia. Virtual Device jäi jumiin koko ajan ja luulin että koodissa oli jotain vikaa. Lopulta kun käynnistin koko koneen uudestaan kaikki toimikin ihan oikein. Uuden activityn luominen ja uuteen näkymään siirtyminen vaikuttivat alkuun hiukan sekavalta mutta taisin sen lopulta tajuta ihan hyvin.

**29.05.2024**

**Harjoitukset 6,7,8.** ChatGPT:ltä sain tähän tehtävään apua muutamassa vaiheessa, kun sovellus ei meinannut toimia. Tietokantojen kanssa puuhailu tuntuu hiukan epävarmalta enkä tiedä olenko sitä ihan täysin ymmärtänyt. Kirjojen järjestäminen numeroiden tai nimien mukaan oli myös yllättävän hankalaa saada toimimaan ja siinä oli myös ChatGPT:stä iso apu, luultavasti olisin luovuttanut ilman sitä.

30.05.2024

**Harjoitukset 6,7,8.** Viimeistelyä. Kaikki harjoitukset sisältyvät samaan AkunTaskukirjat -projektiin.

31.05. – 03.06.2024

**Codelabs harjoituksia:**

**Jetpack Compose harjoitus**: <https://developer.android.com/courses/pathways/android-basics-compose-unit-2-pathway-2>. Halusin hiukan tutustua aiheeseen, koska ilmeisesti Compose on Googlen suosittelema työkalu käyttöliittymien rakentamiseen Androidissa.

**Android Room with a view - Kotlin**: <https://developer.android.com/codelabs/android-room-with-a-view-kotlin#16>. Poikkesin hieman ohjeistuksesta ja tein käyttäen KSP annotaatioprosessia kaptin sijaan, koska sitä suositellaan nykyään käytettävän.

04.06. – 05.06.2024

Katsottu luennot 4 ja 5. Tehty CodeLabsin Firebase- harjoitus (<https://codelabs.developers.google.com/codelabs/firebase-android/#0>)

**Harjoitukset 9 ja 10.** Molemmat harjoitukset löytyvät projektista ’AkunTKFirebase’.

Otin pohjaksi aikaisemman harjoituksen sovelluksen pohjan ja poistin Room kirjaston, jonka korvasin Firebasen tietokannalla. Firebase itsessään oli mielestäni todella helppo ottaa käyttöön ja käyttää.

Edellinen Akun Taskukirja -harjoitus oli tehty käyttäen hiukan vanhoja Android ohjelmointi käytäntöjä, jonka takia Firebase harjoituksen ja edellisen tehtävän koodin yhteensovittamisen kanssa oli hiukan hankaluuksia.

06.06. – 07.06.2024

**Harjoitus 11.** Harjoituksen sai aika pitkälti tehtyä googlen sensoridokumentaation avulla (<https://developer.android.com/develop/sensors-and-location/sensors/sensors_overview>)

Ajattelin aluksi tehdä tehtävän vain muutamaa sensoria käyttäen, mutta sitten kun huomasin että TYPE\_ALL vakiolla sai listauksen kaikista laitteen sensoreista, ajattelin että olisi mielenkiintoista tehdä sovellus, joka listaa niiden kaikkien tiedot.

Koska päätin sisällyttää sovellukseen kaikki mahdolliset sensorit, en viitsinyt alkaa jokaisesta mahdollisesta tapauksesta selvittämään, että mitä dataa ne sisältävät, joten yksinkertaisuuden vuoksi sovellus palauttaa jokaisen sensorin datan muodossa data 1, data2, data3 jne. poislukien nollan palauttavat data-alkiot.

Tämä harjoitus oli kohtuullisen helppo. Vaikeinta oli näkymän luominen Jetpack Composen avulla, sekä tiedon päivittäminen Compose funktioilla.

08.06. – 09.06.2024

**Harjoitus 12.** Laajin ja työläin harjoitus tähän mennessä.

Tehtävä on tehty CameraX CodeLabin pohjalta (<https://developer.android.com/codelabs/camerax-getting-started#0>). Sovelluksessa on jonkin verran eroavaisuuksia CodeLabiin, johtuen mm siitä, että CodeLab-esimerkkiä ei ollut tehty Composea käyttäen. Sovellukseen on myös tehty lisäominaisuutena otetun kuvan näkymä, sekä kamerasovelluksella otettujen kuvien ja videoiden thumbnail listaus.

Lupa-asiat sain tehtyä ChatGPT:n avustuksella, ne olivat (ja ovat edelleen) hiukan sekavan tuntuisia.

Olin kameran ulkonäköön ja toimintaan melko tyytyväinen lopulta. Toiminnallisuuteen olisi voinut tehdä pieniä parannuksia, esim. videota otettaessa jos, kääntää kameran niin sovellus kaatuu.

10.06-11.06.2024

**Harjoitukset 13 ja 14.** Projektin nimi: Askelmittari

Harjoitus 13 oli helppo ja nopea tehdä. Puhelimestani onneksi löytyi tarvittavat sensorit, joten pystyin hyödyntämään STEP\_COUNTERia seuraavassa harjoituksessa.

Harjoitus 14 oli melko haastava. Lähdin alkuun toteuttamaan laskuria STEP\_DETECTORIa käyttäen, jossain vaiheessa kuitenkin kävi ilmi, että STEP\_ COUNTER on tarkempi, joten vaihdoin siihen.

Tämän tyyppiseen harjoitukseen löytyi paljon esimerkkejä googlesta, yksikään ei vaan käyttänyt Composea, joten sovelluksen logiikkaa joutui hieman säätämään.

Käytin myös ChatGPT:tä apuna, ja siitä jäi hiukan kahtiajakautunut fiilis. Toisaalta ChatGPT:stä oli apua SharedPreferencen hyödyntämisessä, mutta se myös loi järjettömän paljon turhaa koodia, jota ei todellisuudessa tarvittu sovelluksen toteutuksessa mihinkään.

Olen tyytyväinen harjoituksen lopputulokseen. Sovellus tuntuu toimivan hyvin ja sovelluksen ulkoasu on siisti.

Harjoituksessa 14 apuna käytettyjä lähteitä:

* <https://montemagno.com/part-1-my-stepcounter-android-step-sensors/>
* <https://medium.com/@huuphuoc1396/android-step-detector-using-sensors-9f1a265a6069>
* <https://developer.android.com/health-and-fitness/guides/basic-fitness-app/read-step-count-data>
* <https://www.geeksforgeeks.org/how-to-build-a-step-counting-application-in-android-studio/>
* https://developer.android.com/training/data-storage/shared-preferences
* ChatGPT

**12.06-13.06.2024**

**Harjoitukset 15 ja 16.** Projektin nimi: GPS

Päätin mennä helpomman kautta tällä kertaa ja tehdä projektin Viewien avulla, koska ne sai hyvin pitkälti esimerkeistä suoraan.

Ajattelin että pelkkien koordinaattien tallentaminen tietokantaan on ehkä hiukan ylimitoitettua, joten päädyin tässäkin tehtävässä käyttämään SharedPreferenceä, joka oli helppo ottaa käyttöön edellisen tehtävän pohjalta. Hiukan ehkä teennäisen oloiseksi meni tiedostojen lisääminen ja poistaminen sen kanssa, mutta onnistui ihan hyvin kuitenkin.

Kartan lisääminen projektiin kävi yllättävän helposti ja nopeasti, mistä syystä päädyinkin hiukan hifistelemään sen kanssa ja lisäsin sinne hiukan lisätoiminnalisuutta. Klikkaamalla markeria saa näkyviin pisteen koordinaatit ja markkeria pitkään painamalla sen pystyy poistamaan sekä kartalta että tietorakenteesta.

**13.06.2024 – 07.06.2024**

**Harjoitus 17.** Projektin nimi: Valuuttalaskuri

Projekti meni jo hyvin pitkälti rutiinilla. Ainoa asia mikä tuotti enenmmän ongelmia oli XMLParserin käyttö. XML datassa siirtyminen halutulle tasolle oli aika vaikea saada toimimaan, kun datassa oli paljon asioita samalla id:llä.

**Harjoitus 18.** Projektin nimi: Ssovellus

Tämäkin projekti sujui jo melko rutiininomaisesti. Muokkasin XML parseria niin että se käyttää DocumentBuilderFactorya, joka oli omasta mielestäni helpompi tapa käsitellä dataa.

**Harjoitus 19.** Projektin nimi: temperatureWidget

Yhdistin tähän harjoitukseen GPS harjoituksen sekä sääsovelluksen. Tein widgetin joka näyttää puhelimen nykyisen sijainnin mukaisen lämpötilaennusteen seuraavalle kolmelle tunnille. Projektilla ei ole mainActivitya ollenkaan, koska se ei pidä sisällään muuta kuin widgetin.

Widgettien kanssa kikkailu osoittautui hiukan kinkkisemmäksi kuin kuvittelin, mutta löydettyäni verkosta hyvät ohjeet sain kuitenkin tämänkin toteutettua. Ensimmäinen projekti meni niin pahasti sekaisin, että jouduin aloittamaan uuden, ennen kuin sain mitään ylipäätään näkymään puhelimellani. Haasteita tuotti saada sijaintitietojen luvan kysyminen toimimaan widgetissä, sekä omasta mielestäni hiukan sekavasti toteutettu ilmatieteenlaitoksen rajapinta.

Jostain syystä omassa puhelimessani widgetin tekstien fonttikoot eivät pysy samana, vaan ne saattavat yhtäkkiä muuttua, jolloin widget näkymä ei näy enää oikein. En keksinyt mistä tämä johtuu.

Harjoituksessa 19 käytetty lähde: <https://developer.android.com/develop/ui/compose/glance/create-app-widget>

**Harjoitus 20.**

**Ks. liite 2**

**Harjoitustyö**

**Harjoitustyön kuvaus: Liite 3**

**Sovelluksen käyttöohje: Liite 4**

# Ajankäyttö

|  |  |  |
| --- | --- | --- |
| Luennot | 8h | katsottu 1,5-2x nopeudella |
| Oheismateriaali | 25h | mm. codelabs harjoituksia |
| Verkkokurssi | 12h | Programming Fundamentals in Kotlin |
| Harjoitus 1 | 1h |  |
| Harjoitus 3 | 2h |  |
| Harjoitus 4 | 2h |  |
| Harjoitus 5 | 5h |  |
| Harjoitukset 6,7,8 | 10h |  |
| Harjoitukset 9,10 | 5h |  |
| Harjoitus 11 | 5h |  |
| Harjoitus 12 | 15h |  |
| Harjoitukset 13,14 | 10h |  |
| Harjoitukset 15,16 | 8h |  |
| Harjoitus 17 | 5h |  |
| Harjoitus 18 | 5h |  |
| Harjoitus 19 | 10h |  |
| Harjoitustyö | 30h |  |

# AI työkalujen käyttö

ChatGPT:tä olen käyttänyt jonkin verran apuna etsimään koodista pieniä syntaksivirheitä. Myöskin ChatGPT:stä on ollut siinä että kurssin GitHubissa olevat esimerkkikoodit käyttävät paikoitellen vanhentuneita käytäntöjä ja ChatGPT:n avulla niitä on pystynyt helpommin muokkaamaan vastaamaan nykykäytäntöjä.

# Viiteluettelo

1. OnePlus 10 Pro puhelimen tekniset tiedot – OnePlus (Suomi). <https://www.oneplus.com/fi/10-pro/specs>
2. OxygenOS 14 – OnePlus (Suomi). <https://www.oneplus.com/fi/oxygenos14>
3. *Sovelluksen luominen ja valmistelu – Play Console Ohjeet* - <https://support.google.com/googleplay/android-developer/answer/9859152?hl=fi>

# Liitteet

## Liite 1: Harjoitus 1

Tehtävän tavoitteena on selvittää vaaditut asiat valitun mobiililaitteen ohjelmointiin liittyen ja kirjata ne ylös. Valitsin laitteeksi oman puhelimeni eli OnePlus 10 Pro 5G:n.

### Laitteen perustiedot

OnePlus 10 Pro 5G 1

Malli: NE2213

Suoritin: Snapdragon 8 Gen 1 Mobile Platform

RAM: 8 Gt

Näytön koko: 6,7 tuumaa

Akun kapasiteetti: 5000 mAh

Kamerat: Etukamera 32MP, Takakamera 48MP+50MP+8MP.

### Laitteen käyttöjärjestelmä

Laitteessa on käytössä OnePlussan kehittämä Androidiin perustuva käyttöjärjestelmä nimeltään OxygenOS 2. Laitteeni OxygenOS versio on 14, joka on uusin saatavilla oleva versio käyttöjärjestelmästä.

### Ohjelmointikielet ja tarvittavat työkalut

OxygenOS perustuu Androidiin, joten sovellusten kehittämiseen voidaan käyttää Java, Kotlin ja C++ ohjelmointikieliä. Android sovellukset paketoidaan APK tai ABB-pakettitiedostoihin, joista jälkimmäistä Google vaatii nykyään käytettävän uusien Google Play -sovellusten kanssa.

Sovellusten kehittämiseen tarvitaan Android Studio-ohjelmistoa, joka on Android käyttöjärjestelmän virallinen ohjelmointiympäristö. Käytössäni tällä kurssilla on Android Studion tämän hetken uusin versio (Android Studio Jellyfish | 2023.3.1).

### Laitteen ominaisuudet

Laitteesta löytyy joitain valmistajakohtaisia sovelluksia kuten OnePlus Store, Community ja Kloonaa puhelin. Lisäksi puhelimessa on esiasennettuna useita Googlen sovelluksia, kuten Play Musiikki ja YouTube.

Puhdas Android tarkoittaa Googlen tekemää androidia ennen kuin se menee laitevalmistajille. Ainakin Suomessa täysin puhtaat Android käyttöliittymät ovat harvinaisia puhelimissa. OnePlussa oli aikoinaan tunnettu siitä, että sen käyttöjärjestelmä oli hyvin lähellä puhdasta Androidia, mutta viimeisimmissä versioissa OxygenOS on mennyt yhä kauemmas puhtaasta Androidista.

Sovelluskauppana puhelimessa toimii Googlen Play Kauppa, johon oman sovelluksen vieminen on tehty erittäin helpoksi. 3

## Liite 2: Harjoitus 20

Etäopetus sopi itselleni varsin hyvin, varsinkin kun kurssin pystyi suorittamaan kesäkurssina. Videot olivat informatiivisia ja ohjelmointiesimerkit sinällään hyviä, mutta ongelmaksi muodostui, se että hyvin iso osa koodista oli jo vanhentunutta johtuen siitä, että videot oli kuvattu jo vuosia sitten. Lisäksi kurssilla olisi hyvä olla parempi tuki Kotlin ohjelmoinnille.

Onneksi aiheisiin liittyen löytyi googlaamalla ohjeita ja esimerkkejä pilvin pimein. Myös erilaisia Android/Kotlin verkkokursseja tuntui olevan runsaasti tarjolla jopa ilmaiseksikin. Kurssin aikana tulikin enemmän hyödynnettyä kurssin ulkopuolisia lähteitä, tehtävien suorituksessa.

ChatGPT:stä oli runsaasti hyötyä virhekoodien avaamisessa, koodin ongelma yleensä ratkesi hetkessä AI:n avulla. Lisäksi vaikeampien ominaisuuksien toteutuksessa ChatGPT:stä oli iso apu. Käytännössä monessa tapauksessa se korvasi googlettamisen, ja tarjosi vastauksen nopeammin ja ilman että joutui selaamaan useaa hakutulosta. Myöskin kommenttien kirjoittamisessa ChatGPT:stä oli apua, jos ne oli sattunut unohtamaan lisätä koodia tehdessä.

Harjoituksista parhaiten jäi mieleen ehkä Akun Taskukirjojen eri versiot. Siinä sai hyvän käsityksen erilaisista tietokannoista kun ensin harjoitus toteutettiin käyttäen SQL:ää ja sen jälkeen Firebasea. Myös erilaisten sensorien testailu ja kamerasovelluksen tekeminen oli mielenkiintoista.

Kurssin työmäärä oli melko suuri. Osa toteutetuista sovelluksista oli hyvin työläitä ja tunteja kului lisätiedon etsintään ja kurssimateriaalin ulkopuoliseen oheisopiskeluun paljon. Harjoitukset voisivat toimia paremmin niin, että osa koodista olisi valmiina ja harjoitukseen toteutettaisiin vain jokin tietty osa-alue tms.

## Liite 3: Harjoitustyö

Päätin tehdä harjoitustyöksi sovelluksen nimeltä KoiraTreffit.

Sovelluksen ideana on olla sosiaalisen median palvelu, jonka kautta eri koiranomistajat voivat etsiä leikki/harrastus seuraa omille koirilleen. Sovellus käyttää tietojen tallentamiseen ja käyttäjien autentikoimiseen Google Firebasea.

Palvelun käyttöliittymä on pyritty tekemään mahdollisimman yksinkertaiseksi, ja kaikki toiminnallisuus löytyy vain yhden tai kahden klikkauksen takaa.

Kun sovellus avataan, käyttäjältä kysytään lupa sijaintitietojen käyttöön, jonka jälkeen käyttäjä kirjautuu sovellukseen käyttäen Google- tai Facebook-tiliään. (Palautettuun versioon toteutettu pelkästään Google-tilin käyttö)

Varsinaisena käyttöliittymänä toimii karttanäkymä, jossa näkyy käyttäjän lähellä olevat koirapuistot ja muut mahdolliset koirien tapaamispaikat, sekä muiden käyttäjien luomat tapahtumat.

Lopullisessa versiossa palveluun kuuluisivat kaikki Suomen koirapuistot, mutta tähän versioon on lisätty vain muutama Tampereen koirapuisto.

Koirapuistoa/Tapahtumaa klikkaamalla käyttäjä näkee tietoja siihen liittyen ja Keskustelu-nappia painamalla käyttäjä voi kirjoittaa viestejä muille käyttäjille ja nähdä muiden käyttäjien kirjoittamia viestejä.

Käyttäjä voi luoda oman tapahtuman painamalla pitkään pohjassa sitä kohtaa kartasta, johon haluaa tapahtuman luoda. Muut käyttäjät voivat kommentoida tätä tapahtumaa.

Idea voisi olla ihan toteuttamiskelpoinen ihan oikeaksi sovellukseksikin, vaikkakin se vaatisi vielä aika paljon kehittämistä. Käyttöliittymä on tällä hetkellä lyöty kasaan enempiä miettimättä eli UX-puolella olisi runsaasti parannettavaa. Karttanäkymä luultavasti kannattaisi vaihtaa käyttämään Google Mapsia. Lisäksi lisätoiminnallisuutta tulisi lisätä, esim. kuvien lisääminen, notifikaatiot, kaverien lisääminen, paikallista minut-nappi yms. Myös kaikkien viestien moderoiminen pitäisi hoitaa jotenkin.

Lopulta kurssin suorittamiseen kesällä liikenevä aika oli sen verran vähäistä, että tämän viimeistellympi lopputulos ei ollut mahdollinen.

## Liite 4: KoiraTreffit sovelluksen käyttöohje

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, ympyrä

Kuvaus luotu automaattisestiAluksi käyttäjän tulee kirjautua Google-tilillään sovellukseen. Tämän jälkeen käyttäjän on sallittava sovelluksen käyttää laitteen sijantia

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisesti

Tämän jälkeen aukeaa karttanäkymä. Klikkaamalla tassun kuvaa näkymään aukeaa popup-ikkuna jossa on tiedot sen kohdalla sijaitsevasta koirapuistosta. Popup-ikkunan saa suljettua klikkaamalla sen päältä.

Kuva, joka sisältää kohteen teksti, kartta, kuvakaappaus

Kuvaus luotu automaattisesti

Klikkaamalla Keskustelu-nappia pääsee näkemään kyseisen koirapuiston keskustelun. Täällä voi myös kirjoittaa ja lähettää omia viestejä keskusteluun.

Kuva, joka sisältää kohteen elektroniikka, teksti, kuvakaappaus, Elektroninen laite

Kuvaus luotu automaattisesti

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti

Kuvaus luotu automaattisestiKäyttäjä voi luoda oman tapatuman painamalla pitkään kartalla kohdassa jossa haluaa tapahtuman sijaitsevan

Kuva, joka sisältää kohteen teksti, kuvakaappaus, Fontti, kartta

Kuvaus luotu automaattisesti

Tapahtumissa on samanlainen chat-alusta kuin koirapuistoissakin. Käyttäjä voi poistaa itse luomansa tapahtuman painamalla ruksia popup-ikkunan oikeassa yläreunassa.

Sovelluksesta pystyy kirjautumaan ulos painamalla oikessa yläreunassa olevaan uloskirjaus-nappia.