**Compose UI**

Saku Karttunen – 1.12.2022

Tein monen tehtävän Composella, sillä pidän Composen tuomasta tavasta kirjoittaa sovelluksia, sillä se muistuttaa enemmän minua web ohjelmoinnista, josta minulla on jo jonkin verran kokemusta. Compose myös vähentää tarvittavan koodin määrää, joka on erittäin hyvä bonus.

Valitsin esiteltäväksi tehtävän 3 – Quiz App. Sillä se oli mukava rakentaa ja pidin prosessista paljon.

Text

Description automatically generated

Tein tehtävää varten oman tiedoston, johon rakensin data class:in kysymyksiin liittyvää dataa varten, jotta kysymyksien dataan olisi helppo päästä käsiksi. Tämän lisäksi kaikki kysymykset saatiin yhteen listaan todella helposti. Tämä helpotti ja nopeutti ohjelmointia funktioiden kohdalla.

Text

Description automatically generated

Main activityssä rakennusvaiheessa, käytin Kotlinin sisäänrakennettua .shuffled() funktiota. Tämä varmisti sen, että kysymykset tulevat aina satunnaisessa järjestyksessä. Se on heti ensimmäisenä, sillä sen tarvitsee vain shufflata kysymykset kerran heti sovellusta avatessa.

Text

Description automatically generated

Composablet oli helppo myös jakaa pienempiin osiin, jotta niitä oli helppo sitten sijoitella näytölle sopivalla tavalla.

A screenshot of a computer

Description automatically generated with medium confidence

Itse kysymysten compose elementissä, aluksi tein kaikista tarvittavista muuttujista remember muotoisia, jotta niiden muutokset aiheuttaisivat uudelleen piirtämisen. UserScore on myös remember muodossa, sillä käytin sitä debuggauksessa teksi elementissä.

Kysymykset myös muokataan ja siirretään omaan listaansa, jotta myöhemmin niihin on helppo verrata.

Text

Description automatically generated

Tämän jälkeen valitaan kysymys jolla aloitetaan indexin avulla. Otetaan tästä objektista tarvittu kysymys elementti ja siirretään se muuttujaan jota teksti elementti käyttää.

Myös vastaukset lisätään omaan listaansa, jotta ne ovat helppoja lisätä button elementteihin.

Ihan lopuksi shufflataan vielä myös kysymysten järjestys, jotta ne ovat eri järjestyksessä melkein joka kerta.

Text

Description automatically generated

Käytin LazyVerticalGridiä, jotta sain kysymykset mukavasti grid muotoon, tässä lazy gridissä oli myös mahdollisuus käyttää argumenttia cells, joka mahdollisti, että rivissä oli aina kaksi elementtiä vierekkäin.

Tämä elementti looppaa sitten vastaukset listan, joille se rakentaa jokaiselle oman button elementin sekä onClick funktion.

Funktiossa käytin guard clause tapaa, jolla ohjelma tarkistaa, että onko kysymyksiä vielä jäljellä ja sitten vaihtaa kysymyksen ja palaa takaisin. Jos ei löydy toista kysymystä, laukaistaan sendResults funktio.

Text

Description automatically generated

sendResults() funktio tekee puolestaan intentin, johon sisällytetään käyttäjän pistemäärä sekä oikeat vastaukset.

Text

Description automatically generated

Result näyttö on erillinen tiedosto ja erillinen compose kokonaan. Siellä näytetään vain tulokset numeroina ja luodaan nappula, jolla voidaan palata ensimmäiselle sivulle, jotta voi pelata uudestaan.

Tämä nappula ei palauta dataa Intentin mukana, jotta joka kierroksella peli alkaa puhtaalta lautaselta.

**Mitä opin tehtävän aikana:**

**-Kuinka käyttää intenttiä Compose projektissa.**

**-Data classien rakentamisen Kotlinissa**  
Tämä oli erittäin helppoa, sillä classiin ei tarvinnut erillistä constructoria taikka muuttujia erikseen kirjoittaa classin sisälle.

-Gridin käyttöä composessa

-Että listojen ”shufflaaminen” on huomattavasti helpompaa kuin JavaScriptissä, jossa funktio piti rakentaa itse

-Kuinka käyttää remember muuttujia

-Paljon muuta pientä korjailua ja Kotlinin vahvuuksien hyväksi käyttämistä.