Deckblatt

Gliederung

Einführung

Die Android Applikation „Rocket League Companion“ ist ein Projekt für das Modul „Mobile und Verteilte Systeme“.

Die Zielgruppe besteht aus Spielern des Videospiels „Rocket League“, die sich für ihre eigene Spielstatistik und Entwicklung interessieren. Die App zeigt dem Nutzer sämtliche relevanten Daten, die aus dem Spielerprofil zu ermitteln sind. Das Hauptaugenmerk liegt dabei auf der Entwicklung des einzelnen Spielers und seiner persönlichen Statistik.

Was ist *Rocket League?*

Rocket League ist ein sehr schnelles Multiplayerspiel, bei dem der Spieler ein Auto steuert und versucht den Spielball im gegnerischen Tor unterzubringen. Gespielt wird 3vs3, 2vs2 oder 1vs1 auf einem mit Fußball vergleichbarem Spielfeld. Allerdings mit Wänden und einer Decke.

Die Autos sind mit einem Raketenantrieb ausgestattet und können in der Luft frei gesteuert werden. Dadurch können geübte Spieler kurze Strecken durch die Luft fliegen und präzise Schüsse und Pässe abgeben.

Mit „Rocket League Companion“ kann der Nutzer seinen Spielstyl analysieren, seine Trefferrate und sein Ranking Verlauf betrachten und daraus Schlüsse bezüglich seines Trainings oder anderer äußeren Umstände ziehen.

Überblick

[APPINTRO]

Beim Erststart erhält der Nutzer eine kurze Übersicht über die Funktionen und Anzeigen der App

[SPIELERSUCHE]

Mithilfe der Suche kann der Spieler sein Profil finden und Auswählen

[PLAYEROVERVIEW]

Auf der Hauptseite findet der Spieler rohe Daten zu sich, wie Ranking und Statistiken

[SATISTIC]

Hier findet der Spieler einen Verlauf seiner Trefferrate an diesem Tag und insgesamt, sowie seines Rankings in den verfügbaren Warteschlangen. Zuletzt gibt es ein Kuchendiagramm, an dem er seinen Spielstyl erkennen kann.

Frameworks

Volley

Im Android Projekt „Geonet“ (erhältlich im Playstore) habe ich das Framework Volley kennengelernt. Mit ihm lassen sich einfach kleine Daten Mengen mit http Requests anfordern und verarbeiten. Ich habe es wegen seiner sehr kompakten Schreibweise und einfachen Verwendung dem vorgeschlagenen „OKHTTP“ vorgezogen und vor allem weil ich damit schon gute Erfahrungen gemacht habe und keine Probleme hatte.

Glide

Ich verwende Glide, um die Profilbilder der Nutzer zum einen in der Suche und zum anderen auf der Profilübersicht anzuzeigen. Es vereinfacht dies extrem, da sowieso zu jedem Spieler eine Avatar URL vorhanden ist und ich so mit einer Zeile die URL auslesen und mit einer weiteren Zeile das Profilbild aktualisieren kann.

MPAndroidChart

Anstelle von AChartEngine. MPAndroidChart ist ein immernoch weiterentwickeltes Framework um Graphen und Diagramme anzuzeigen. Für meine App war es essentiell einen Verlauf von Werten anzuzeigen, deshalb habe ich zwingend eine Graphen anzeige benötigt. Auch wenn das Styling und Initialisieren etwas umständlich ist, war es doch die beste Lösung für mein Anwendungsfall, die ich finden konnte.

Nicht verwendete Frameworks

Butterknife

Ich verwende Butterknife nicht, da dich die App in Kotlin implementiert habe und dort direkt auf UIObjekte zugegriffen werden kann, nachdem sie importiert wurden. Die Sprache beinhaltet also praktisch alle Vorteile, die dieses Framework mit sich bringt.

Icepick

Durch seinen simplen Aufbau und das Fehlen von Eingabefeldern ist ein Speichern des aktuellen States nicht notwendig. Sämtliche heruntergeladenen Daten werden lokal per Objektoutputstream direkt nach dem Parsen gespeichert. Leider ist Icepick außerdem nicht mit Kotlin einfach verwendbar.

Funktionsweise

Suche

Die Suche funktioniert simpel. Der Nutzer gibt einen Suchbegriff ein, der dann per passendem Request an den API-Server gesendet wird. Dieser Antwortet dann mit einem JSONArray das Spieler beinhaltet die der Server für passend erachtet. Daraus generiert die App Player Objekte, die dann in der ListView angezeigt und ausgewählt werden können. Durch Antippen eines Spielers wird dieser gesetzt und man gelangt in den Statistikbereich.

Spielerübersicht

Nach der Auswahl wird direkt eine Anfrage an den Server gestellt, um Details zum Spieler zu ermitteln, sie wird dann in einem festen Zyklus wiederholt.

Die Daten werden dann zum einen direkt im Spieler und zum anderen in einem Timestamp Objekt gespeichert, dass sämtliche wichtigen Werte und eine Zeitmarke enthält. Der Spieler hat eine Liste mit Timestamps, allerdings für jeden Tag genau einen. Ist bereits einer für diesen Tag vorhanden wird er überschrieben. Dies verhindert verwirrende Extremwerte, wenn der Spieler zum Beispiel in mehreren Spielen hintereinander 100% Trefferrate hatte.