GitHub Username: flaiker

SC2 Profiler

Description

Show profiles and ladder rankings of StarCraft II

Intended User

StarCraft II players and eSports enthusiasts

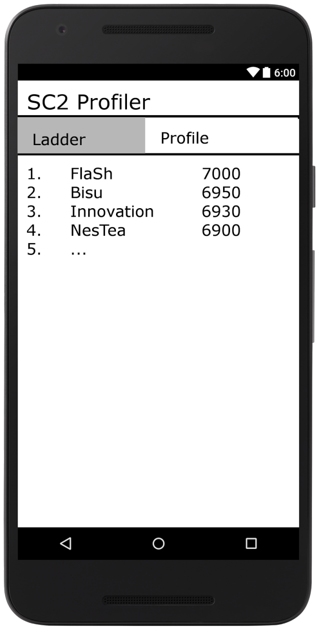
Features

- Display Grandmaster ladder rankings of each SC2 region
- Display your own SC2 profile (using battle.net OAuth login) or any profile you know the id of

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

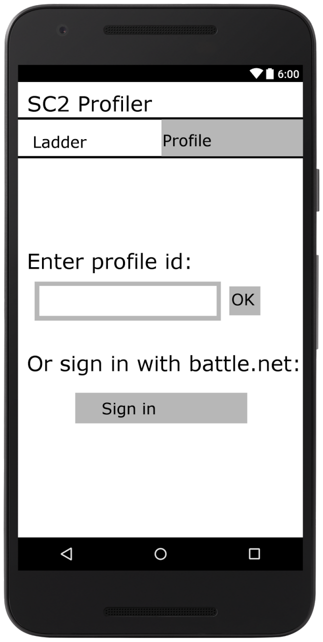
Screen 1



1.	FlaSh	7000
2.	Bisu	6950
3.	Innovation	6930
4.	NesTea	6900
5.	...	

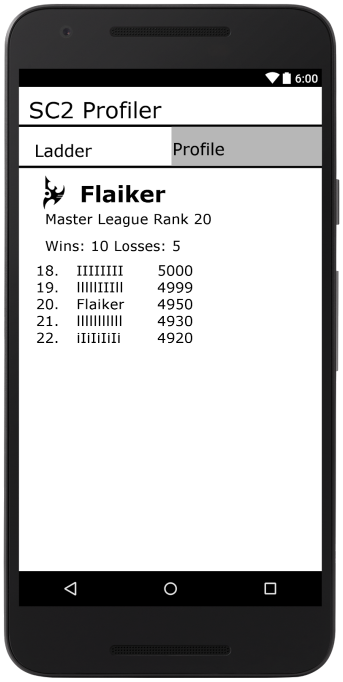
Main screen ladder tab for global GrandMaster ladder rankings

Screen 2



Profile tab, possibility to show profile by id or by logging in via battle.net OAuth

Screen 3



SC2 Profiler

Flaiker

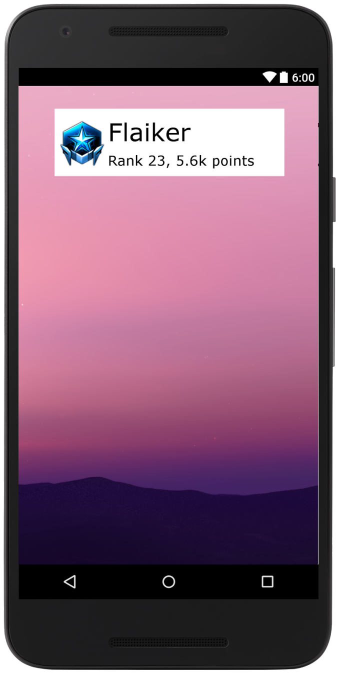
Master League Rank 20

Wins: 10 Losses: 5

18.	IIIIIIII	5000
19.	lllllIIIIl	4999
20.	Flaiker	4950
21.	lllllllllll	4930
22.	iIiIiIiIi	4920

Profile detail view

Homescreen widget



Flaiker

Rank 23, 5.6k points

Shows currently logged in user’s rankings

Key Considerations

How will your app handle data persistence?

New content provider for caching web service requests

Describe any libraries you'll be using and share your reasoning for including them.

- If it works well https://github.com/dementhius/battlenet-oauth2 for authentication
- Picasso for image loading / caching

Describe how you will implement Google Play Services.

- Analytics to see if the amount of app usage correlates with eSport events
- AdSense for monetization

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Set up new Android project
- Initialize Git repository
- Add gradle dependencies

Task 2: Implement UI for Each Activity and Fragment

- Build MainActivity UI
    - Tabbed layout
- Build UI for ladder fragment
- Build UI for profile fragments

Task 3: Implement ContentProvider

- Add a new ContentProvider
- Tables:
    - Ladders
    - Profiles
- Implement services to provide content from https://dev.battle.net/ to the content provider
- Implement a SyncAdapter to handle regular synchronization for the global ladder

Task 4: Bind data to the UI

- Show data of the ladder table in the ladder fragment
    - Use a Loader to do so
- Show requested data in the profile fragment
    - Use a Loader to do so

Task 5: Set up battle.net OAuth

- Implement OAuth login flow to retrieve the user’s profile ID
- Use the ID to show the profile in profile fragment