

System design document for Ripped

[Introduction](#)

[Design goals](#)

[Definitions, acronyms and abbreviations](#)

[References](#)

[Proposed system architecture](#)

[Software decomposition](#)

[General](#)

[Tiers](#)

[Communication](#)

[Layering \(anpassas efter indelning, tex MVC\)](#)

[Dependency analysis](#)

[Concurrency issues](#)

[Issue handling](#)

[Persistent data management](#)

[Access control and security](#)

[Boundary conditions](#)

[References](#)

[Build Procedure](#)

[Release Procedure](#)

Introduction

Ripped is an Android application which will let you log your exercises from the gym. You can create your own workouts with exercises already from the database or create your own exercises.

Design goals

An application with good looking and uniform design so that the look does not differ depending on which android version the user has.

Easy way of handling workouts and exercises, editing workouts (even during a workout) and easy to input results.

Definitions, acronyms and abbreviations

WorkoutTemplate

A workouttemplate is a routine where the user can add and/or remove the exercises that would be performed during the visit to the gym.

Workout

A workout is created when the user has performed a workouttemplate. A workout is used to display the history.

Exercise

Something the user will be doing at the gym. There are three types; cardio, dynamic and static.

Cardio exercise

A cardio exercise could be running, row or cycling. Will allow the user to input the duration of the exercise and the distance.

Dynamic exercise

An exercise where the user can enter number of repetitions and the weight. For example, bench press, military press or dead lift.

Static exercise

An exercise such as the plank bridge or side bridge. This type of exercise will allow the user to input duration and weight.

Reps

Short for repetitions.

Sets

A set contains either the repetition and weight, time and weight or time and distance. So when the user has finished his bench press session of 10 reps with X kg, that will complete a set. The next number of reps and weight X a new set has been performed.

References

N/A

Proposed system architecture

The android application will show both some example exercises and workouts, and also the exercises and workouts created by the user.

This data will be fetched from a local content provider (SQLite Database).

The main design pattern will be the model-view-controller.

Software decomposition

General

The application is divided up in three parts which is divided up in more sub parts.

The three major parts is Model, Controller and View.

The view is what the user will see and use. The view is divided up in more subparts, one for each part of the GUI.

The model is our local SQLite database which contains several tables for storing workouts workouttemplates, exercises, sets, profile data etc.

The controller is also divided up in sub parts depending on which GUI will use them. The controller lets the view parts to write and get data to and from the database.

Tiers

Only the application.

Communication

No communication from the phone to the outside world

Layering (anpassas efter indelning, tex MVC)

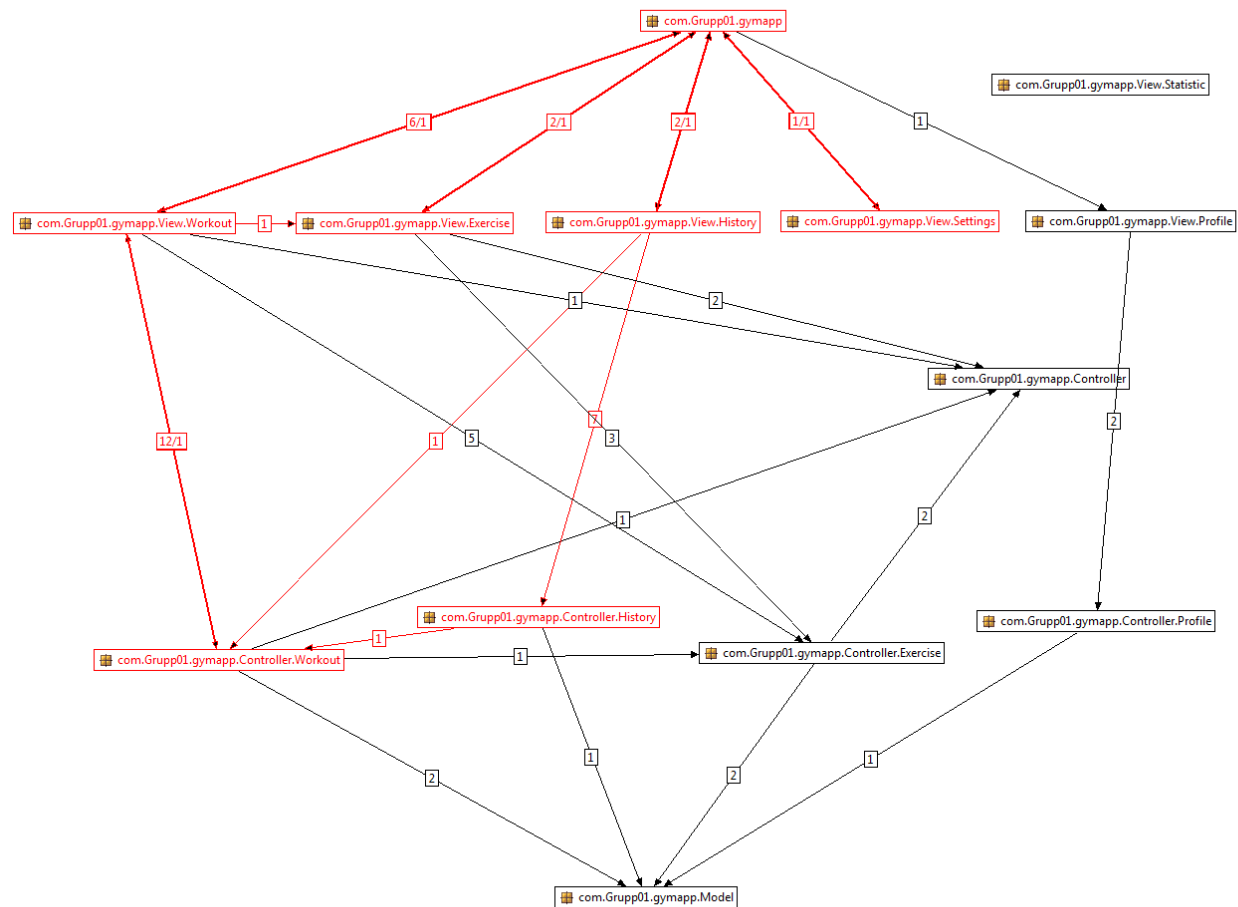
The application is divided up in three layers and is designed with the MVC model in mind.

The model is our local SQLite database.

The controller is classes with method that will access the database and return the data.

The view is the applications activities (and some help classes) and is what the user will see. The view will fetch data from the model via the access methods in the controller classes.

Dependency analysis



Concurrency issues

Everything runs in the same thread, since there is not so much data that is fetched from the database there is no problem so far. However, there may be a need to move that to a separate thread if we notice that there is a big delay between opening an activity and the data is displayed in the activity.

Issue handling

As soon as an issue or a bug has been discovered, it has been added to 'Issues' in our github. Everyone in the group has regularly been checking this and if someone nows how to solve the issue, they have created a new branch, fixed the issue and then merged the branch into the master.

Persistent data management

The user selected settings (such as language and units for weight and distance) is managed by Android default way of handle this. By creating a `SettingsActivity` the system itself will manage these in the `SharedPreferences` storage.

All other information, such as workout templates, exercises, sets for exercises etc is stored in a local `SQLite` database.

Access control and security

Since all the activity in the application is on the local phone (no central server) and there is no sensitive information being stored there is no access control or security implemented.

Boundary conditions

N/A

Build Procedure

Branches which are finished are merged into the master branch. When test cases are finished we compile it (using Eclipse) and bugs are reported as mentioned above.

Release Procedure

Before implementing any new feature or a bugfix, the feature has had its own branch, where all the developing has taken place. Before merging this feature or bugfix into the master branch it should pass JUnit, Robotium and acceptance tests, after that the branch was built and pushed to git together with a document containing release notes.

References

developer.android.com

actionbarsherlock.com/