Keep Fit

Martin Tiggerdine

Stages Attempted

I attempted every stage:

- Showing the current day status
- Managing a number of goals
- Keeping a persistent history of daily information
- Allowing historical recording of activity on any past date
- Supporting user preferences and settings
- Notifying users when 50% of their day’s goal has been met (extra)
- Supporting automatic recording of activity using a step counter (extra)

App Design

I tried to comply with the Android Developers Core App Quality Guidelines and Material Design Guidelines.

With respect to core app quality, I complied with as many criteria as possible:

- Standard design (the app follows Android Design guidelines and does not mislead or confuse readers)
- Navigation (the app supports Back button navigation, all dialogs are dismissible using the Back button, pressing the Home button navigates to the Home screen)
- Notifications (the app uses notifications only to indicate a change in context relating to the user personally)
- Permissions (the app requests only the absolute minimum permissions that it needs (none))
- UI and Graphics (the app supports both landscape and portrait orientations)
- User/app state (the app does not leave any services running when the app is in the background, the app correctly preserves and restores user or app state)
- Stability (the app does not crash, force close, freeze, or otherwise function abnormally)
- Performance (the app loads quickly)
- SDK (the app runs on the latest public version of the Android platform, the app targets the latest SDK to minimize the user of any compatibility fallbacks)
- Visual quality (The app displays graphics, text, images and other UI elements without noticeable distortion, blurring or pixilation, the app displays text and text blocks in an acceptable manner)
- Data (All private data is stored in the app's internal storage, no personal or sensitive user data is logged to the system or app-specific log)
- Libraries (all libraries, SDKs , and dependencies are up to date)
- Execution (the app does not dynamically load code from outside the app's APK)

I used some of the test procedures too.

Material Theming helped me to customise things like colour and typography. For example, I used the Material palette generator to generate a usable and aesthetically pleasing palette.

I used components like the top app bar, a couple of floating action buttons, dialogs, a list, a menu and tabs and complied with the principles of each. For example:

- the top app bar is persistent (it appears at the top of each screen and can disappear upon scroll), guiding (it provides a reliable way to guide users though the app) and consistent (it has a consistent position and content to increase familiarity)
- the floating action buttons are primary (they represent the primary actions on the screens), constructive (they perform constructive actions (such as record activity or create goal) and contextual (they are relevant to the screens on which they appear)

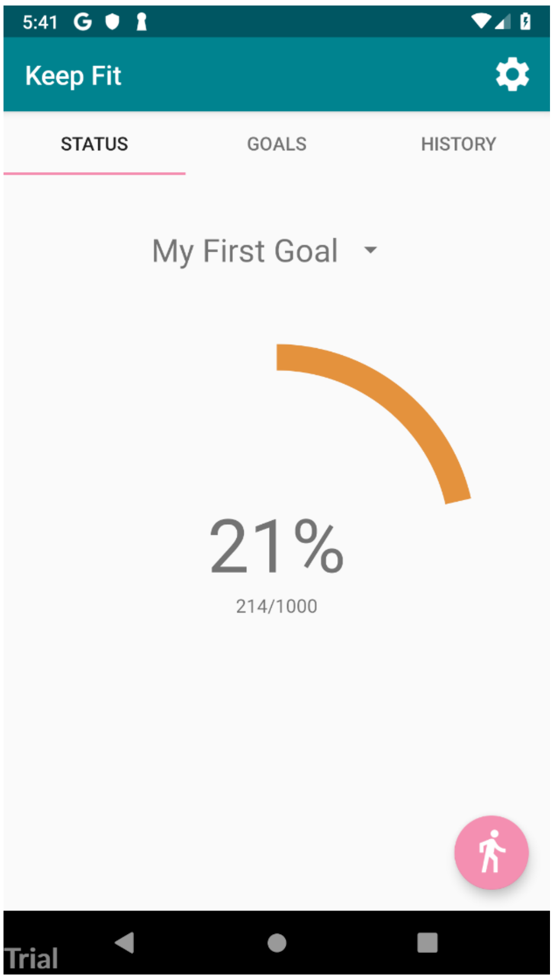
- etc.

I adopted some ideas from similar apps like

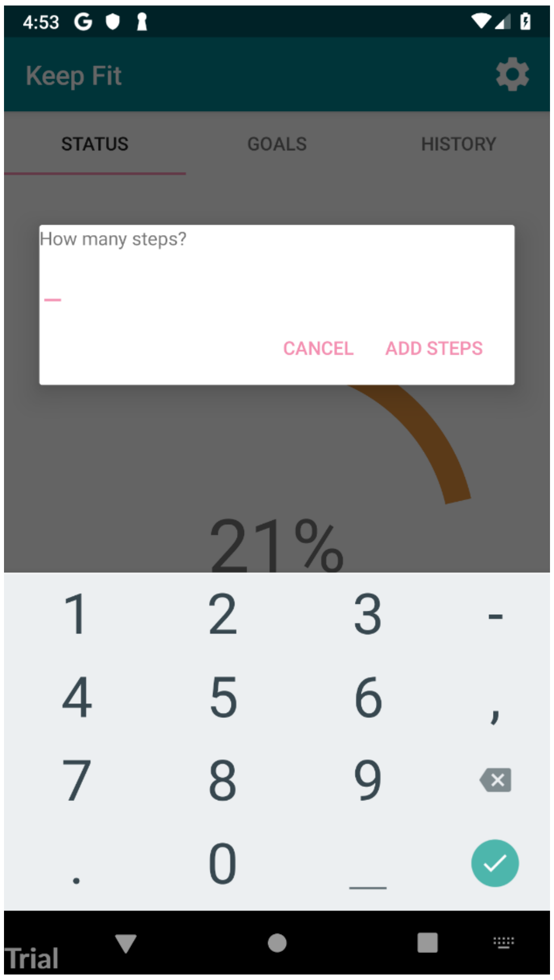
- Circular progress bar
- Floating action button

Implemented Functionality

Showing the current day status



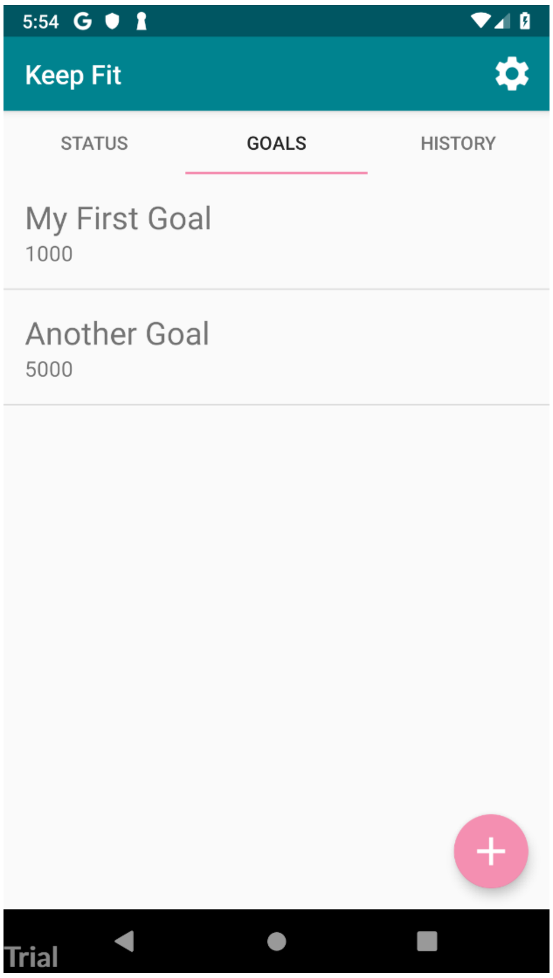
Screenshot 1: Showing the current day status



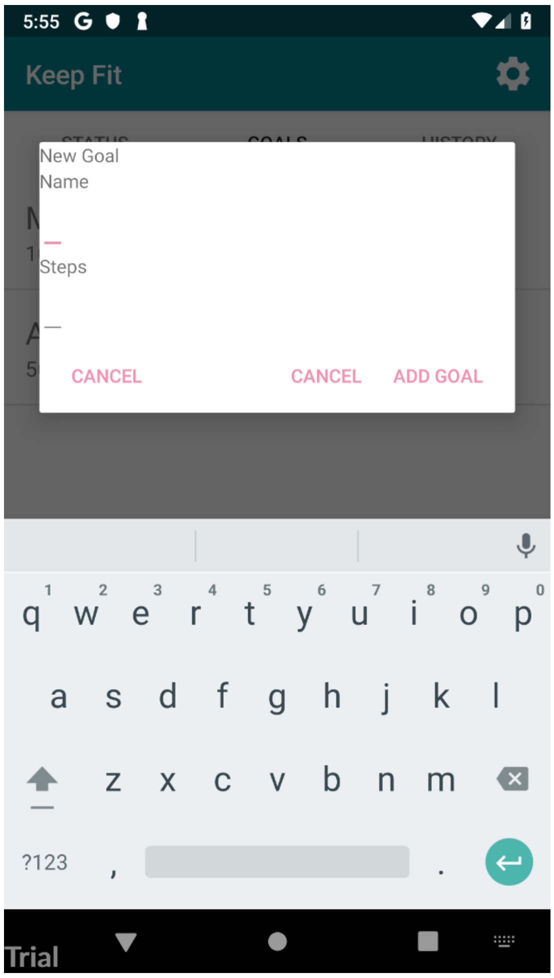
Screenshot 2: Recording activity

The current day status (Screenshot 1) is shown via four views: a Spinner that shows the selected goal of the day (My First Goal), a ProgressWheel that shows the proportion of the goal completed graphically and two Text Views, one that shows the proportion of the goal completed in text (21%) and one that shows the goal's target steps and the activity recorded (214/1000). The FloatingActionButton in the bottom right corner opens a Dialog (Screenshot 2) where the user can record activity. The ProgressWheel and TextViews are refreshed whenever activity is recorded and the colour of the ProgressWheel changes from red to green depending on how close the user is to completing their goal. The user can navigate the app either by swiping anywhere on the screen or by clicking one of the three tabs.

Managing a number of goals

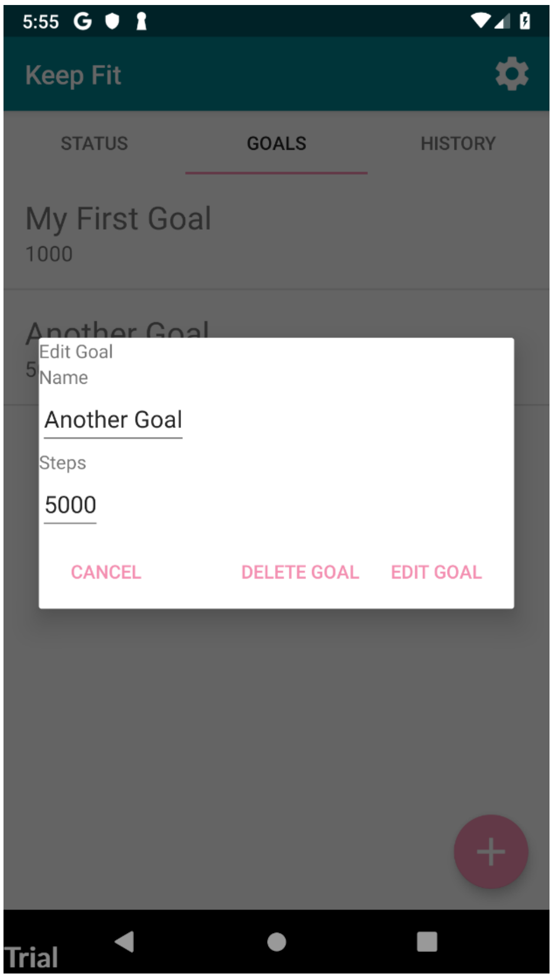


Screenshot 3: Managing a number of goals



Screenshot 4: Allowing goals to be added

Goals can be added (Screenshot 4), edited and removed (Screenshot 5). The goals themselves are stored in a Room and shown using a ListView and an adapter (Screenshot 3). I know I could have used a RecyclerView but I used ListView because it was covered in the Udacity tutorials I used to learn how to develop Android apps. RecyclerView would perform better and be more flexible, especially if there were hundreds or thousands of goals, but there are not likely to be more than about a dozen.

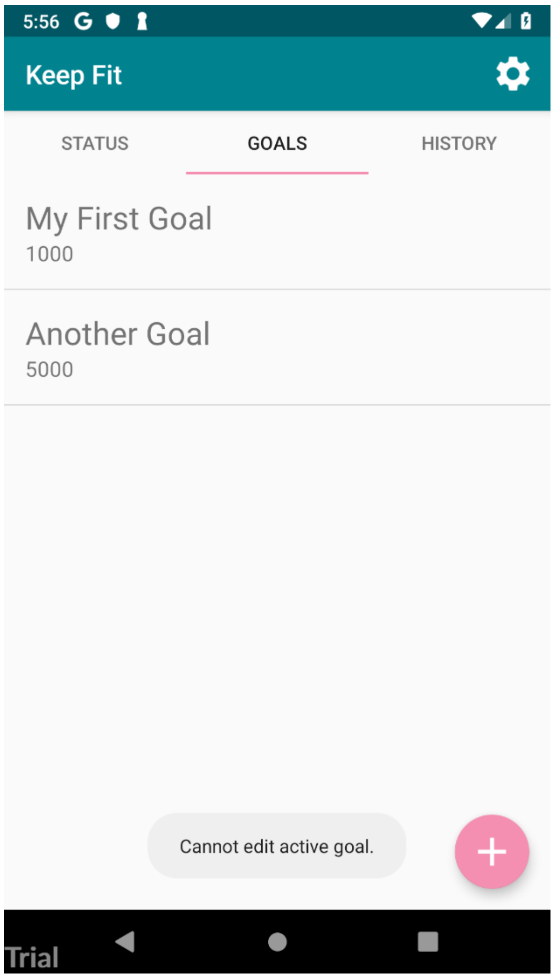


My First Goal
1000

Another Goal
5

Edit Goal
Name
Another Goal
Steps
5000

Screenshot 5: Allowing goals to be removed and edited



My First Goal
1000

Another Goal
5000

Cannot edit active goal.

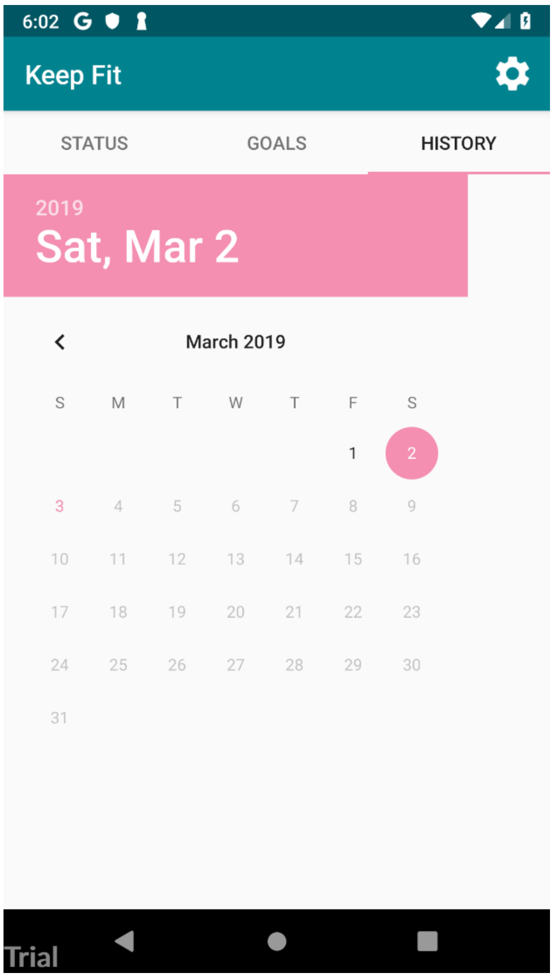
Screenshot 6: Goals can only be edited and deleted when they are not active

Some input validation is done whether the user is adding or editing a goal. Adding or editing is unsuccessful if:

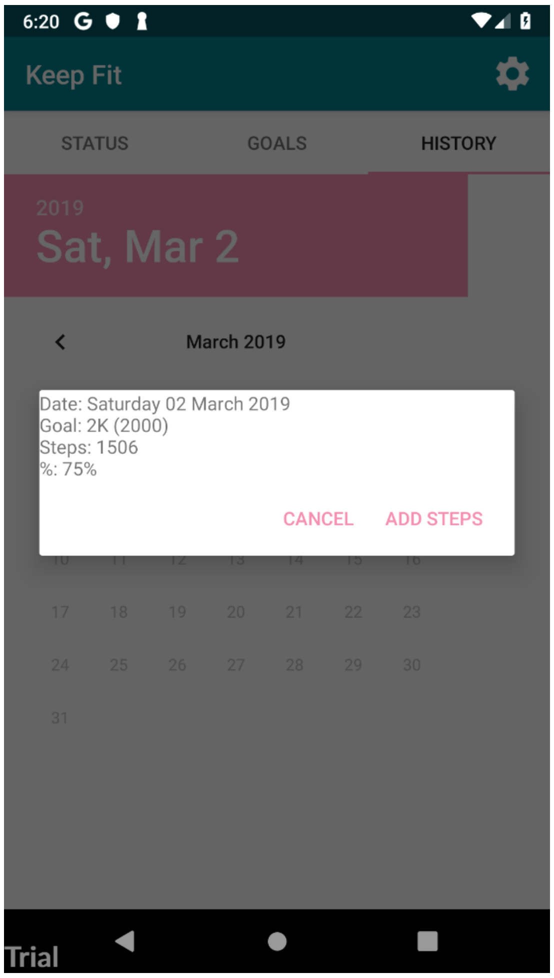
- the name is blank or whitespace
- the number of steps is blank or 0
- there is another goal with the same name
- goal editing is disabled in the settings

The active goal cannot be edited or deleted (Screenshot 6).

Keeping a persistent history of daily information



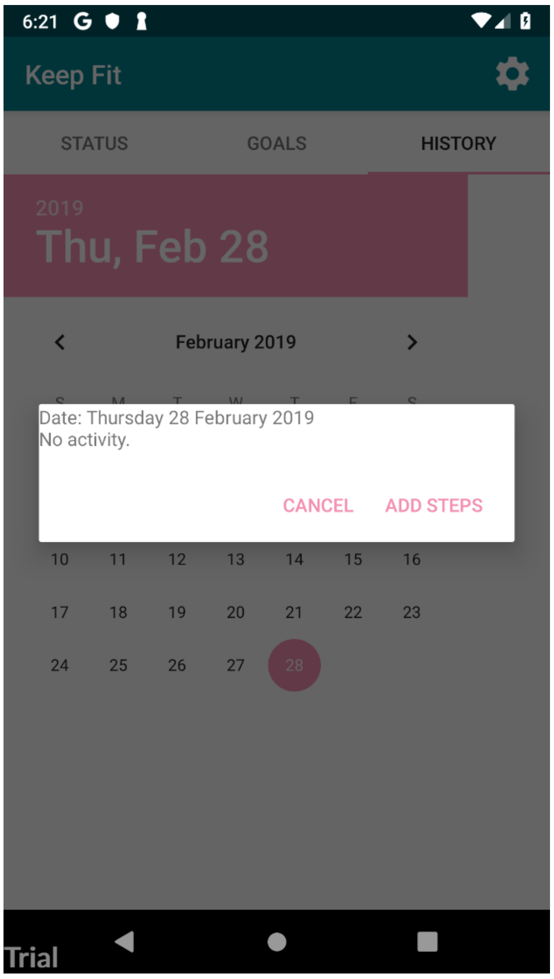
Screenshot 7: Keeping a persistent history of daily information



Date: Saturday 02 March 2019
Goal: 2K (2000)
Steps: 1506
%: 75%

Screenshot 8: Including the goal of the day, the recorded activity and the proportion of the goal completed

A persistent history of daily information is kept in a different table of the same Room used to store goals. There is one entry per day (the date is the primary key). The history is accessed via a calendar DatePicker (Screenshot 7) with current and future dates greyed out. Picking a date opens a Dialog that shows the date, the goal of the day, the recorded activity and the proportion of the goal completed.



6:21

Keep Fit

STATUS GOALS HISTORY

2019

Thu, Feb 28

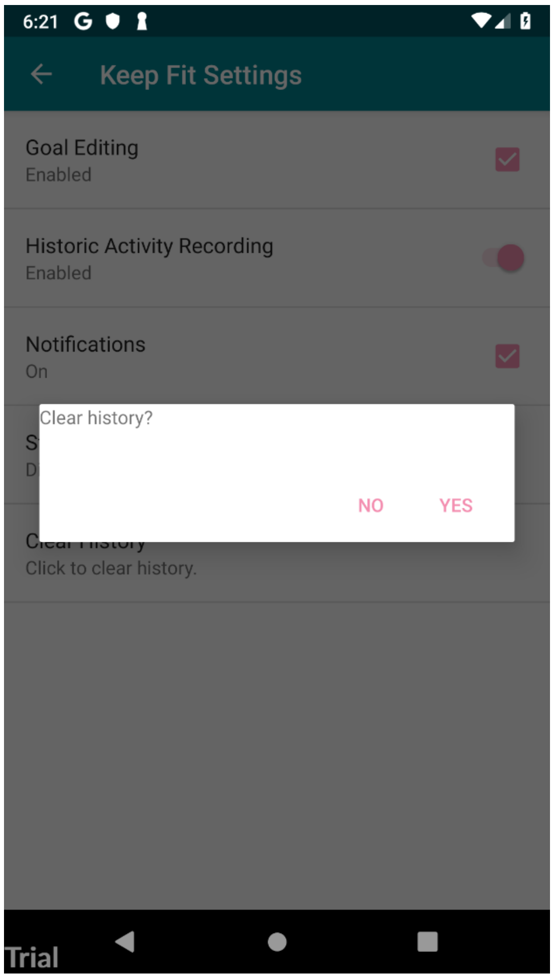
February 2019

Date: Thursday 28 February 2019
No activity.

CANCEL ADD STEPS

Trial

Screenshot 9: Editing the history is not possible



6:21

Keep Fit Settings

Goal Editing
Enabled ✓

Historic Activity Recording
Enabled

Notifications
On ✓

Clear history?

NO YES

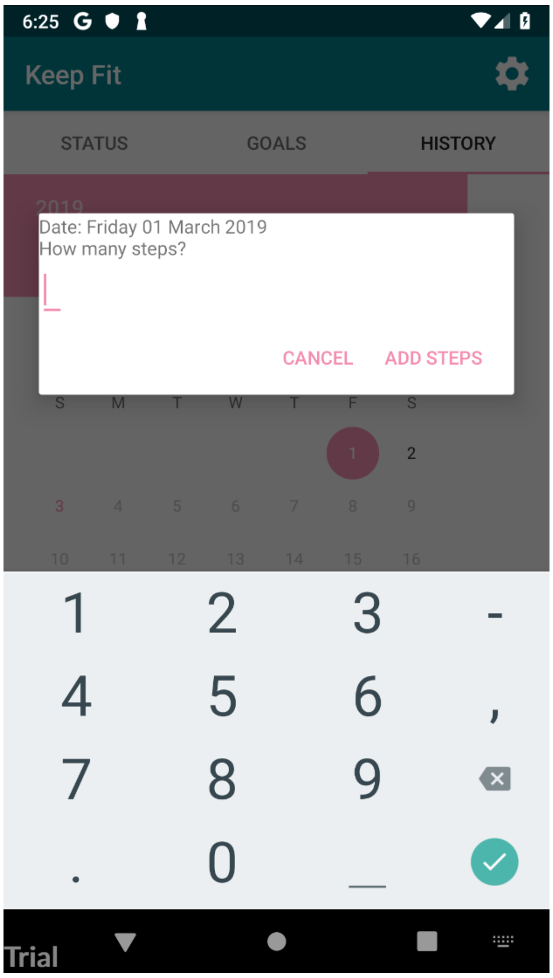
Click to clear history.

Trial

Screenshot 10: Clearing the whole history is supported

If there was no activity that day, the Dialog shows the date and “No activity” (Screenshot 9). Clearing the history (Screenshot 10) is possible by clicking the “Clear History” button in the Settings (see “Supporting user preferences and settings”).

Allowing historical recording of activity on any past date

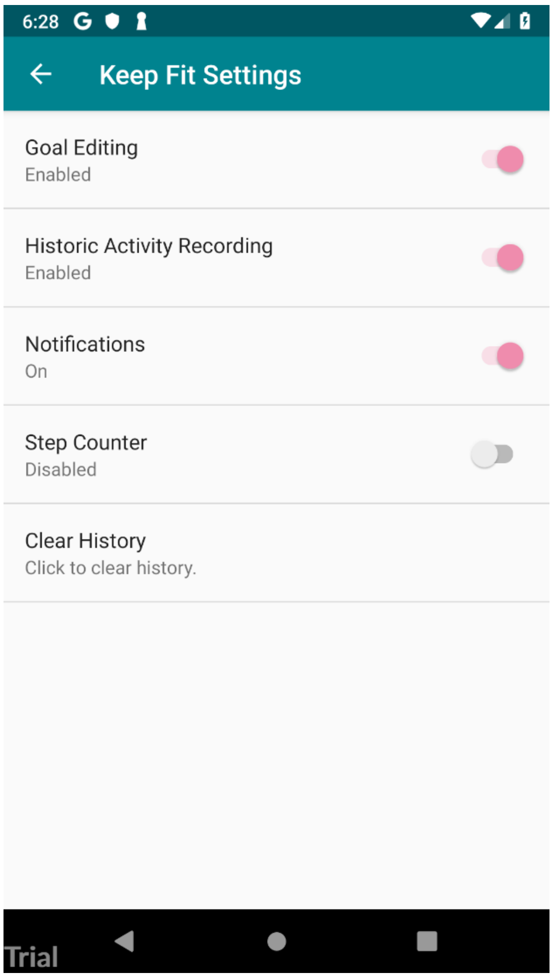


Date: Friday 01 March 2019
How many steps?

Screenshot 11: Allowing historical recording of activity on any past date

Clicking the “Add Steps” button on a history Dialog allows the user to retroactively record some activity on that day (Screenshot 11). The number they input is added to the number in the database so no activity already recorded is lost. The chosen date is made clear in the Dialog.

Supporting user preferences and settings


6:28

Keep Fit Settings

Goal Editing
Enabled

Historic Activity Recording
Enabled

Notifications
On

Step Counter
Disabled

Clear History
Click to clear history.

Trial

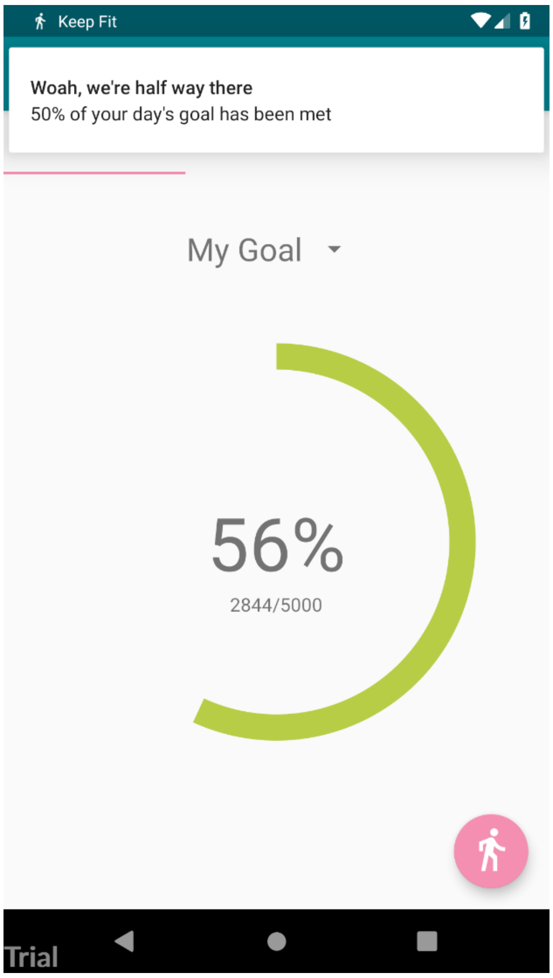
Screenshot 12: Supporting user preferences and settings

Clicking the settings button in the top app bar opens the settings (Screenshot 12). The options are:

- enable/disable goal editing
- enable/disable historic activity recording
- switch notifications on/off
- enable/disable step counter
- clear history

Each option (except clear history) is a switch that can be toggled on or off. If goal editing was disabled, the “Enabled” below “Goal Editing” would change to “Disabled” to reflect this.

Notifying users when 50% of their day’s goal has been met (extra)


Keep Fit

Woah, we're half way there

50% of your day's goal has been met

Screenshot 13: Notifying users when 50% of their day’s goal has been met

Whenever activity is recorded, the app checks whether the proportion of the goal completed changes from <50% to >=50%. If it does, a Notification is built. The user is notified that, in the words of Bon Jovi, “Woah, [they’re] half way there”. This can be disabled in the settings. The user is not “spammed” with notifications (they only way they get more than one a day is if they change to a bigger goal).

Supporting automatic recording of activity using a step counter (extra)

The only Android phone I had access to (not my own) did not have a TYPE_STEP_COUNTER Sensor so this feature is untested. I tried my best to use what I learned here to implement it. The main Activity implements the SensorEventListener. It listens to the step counter (if there is one) and overrides the onSensorChanged() method to count one step. I hope my attempt is worth some credit.

Feedback Consideration

Good coverage of similar apps, focusing on UI design of different aspects, but no mention of GUI components identified.

App design provided using some tool, but focus is only on day status screen (no goal management and settings). Some references to GUI components to be used.

Although supporting change of the day through a button may be good for historical activity recording, it's probably not a good idea for showing history, as navigating through history will be a problem.

For changing the day's goal a picker rather than a button may be more appropriate. For navigation to the goals screen swipe tabs or bottom navigation bar may be better.

I took this feedback into consideration by:

- researching and using GUI components like:
  - top app bar
  - floating action buttons
  - dialogs
  - list
  - menu
  - tabs
- developing goal management and settings
- using a calendar DatePicker to solve the problem of navigating through the history
- using a Spinner rather than a button for changing the day's goal
- using screen swipe tabs for navigation