YAAA (Yet Another Alarm App)

GitHub Username: Nulleye

YAAA (Yet Another Alarm App)

Description

A highly customizable “wake up” focused alarm app that will help you to never nod off again or keep you from waking up with a shock. Wake up with a smile every day!

Intended User

This app is for anyone who needs help to get awake on time, or want a pleasant or “every day different” wake up experience.

Features

A highly customizable “wake up” focused alarm app that allows you to define an arbitrary number of alarms and has these main configuration options, for each individual alarm:

- Dismiss alarm modes:
    - **Default**: swipe left to Snooze, or swipe right to Stop.
    - **Shake**: phone shake or physical button push to Snooze, or Push a screen button (in a random position) to Stop.
    - **Conscious**: No snooze option and a complex “connect the dots” or some small quiz to Stop.
- Set alarm repetition:
    - **None**: next planned day based on the selected time (today or tomorrow), or allows to select a determined calendar day.
    - **Week days**: all week days appear an allow to select all or some of them.
    - **daily, weekly, monthly, annual**: requires to select a determined calendar day.
- Set a number of “are you really awake?” checks in a determined period of time.
- Set up a maximum volume and a gradual volume elevation interval.
- Set an alarm ringtone or a determined song, album or artist to play locally, or from a streaming service like Spotify, Last-fm, Shoutcast radios or Google Play Music **(*)**.
- Set up an auto-delete option: when an alarm has no more planned occurrences or a determined day has passed, the alarm will simply delete itself.

General configuration

Vacation period: it will allow to set a “dismiss all alarms” in a period of time (for holidays, large weekends etc.), except for un-dismissible alarms (a special alarm check).

Global settings and default alarm options: set alarm default options, if they are not explicitly specified on each alarm like, the Dismiss alarm mode, default “are you really awake?” tries, default maximum volume and gradual volume elevation interval, the default alarm sound to use, and the notification and snooze intervals (these last settings are global only).

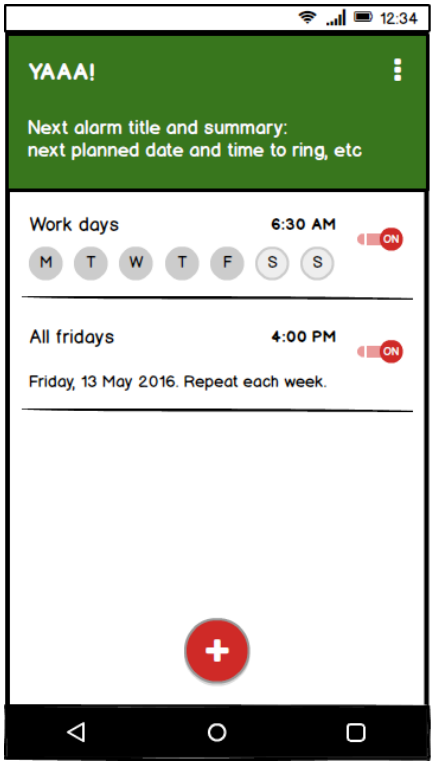
For Udacity project reviewer

(*) These are more a wish, or a future plan, than the actual streaming services it will connect to, as some are quite difficult to implement or require a quite complex selection interface screen (I only have one month and I have a job to attend).

- **Spotify**: I will try to connect to this one, as I have used it in the Spotify streamer projects, but only If I have enough time to do it. It has an official Android SDK but is in beta right now.
- **Last-fm**: it is very similar to Spotify; in fact, it internally uses Spotify streaming. It has an official Android library.
- **Shoutcast**: it is a very good, and a quite simple to implement, streaming service but a Dev partnership and a DevID needs to be requested (I’m waiting for a response). However, streaming metadata and navigation may be tedious to implement as I think there is no Android lib.
- **Google Play Music**: it is the one I would like to connect the most, as I use it every day, but an official Google API has not yet been released (and I think they don’t plan to). There is an unofficial API though, but this is risky, and I hear about a way to use it by using a special functionality for Android TVs, but I’m not sure if it works, any advice will be very appreciated here.

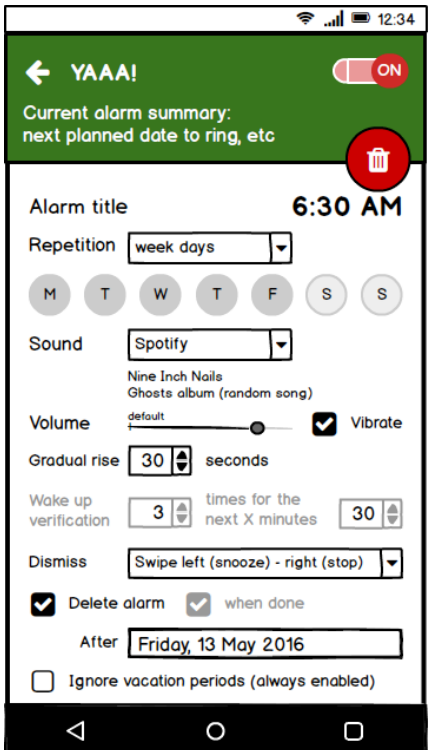
User Interface Mocks

Screen 1 - Main



This is the **Main** application screen.
It complies with the Material design principles and uses a **coordinator layout scrolling** effect and an add alarm **floating button**.
When expanded, the title area shows the **next planned alarm** title and its summary information.
All alarms are listed below, with the alarm title, the time, the planned date summary, and the **On-Off** switch, the only active element for each list item.
Alarms are listed in “time within the day” order from sooner to later, ignoring the planned date, or if they are enabled or not.
Clicking on an alarm or adding a new one shows up the alarm **Detail** screen (see screen 2).
The **tree dots** icon gets access to the application menu and to the **Settings** (see screen 3) and **About** (see screen 6) screens.
Alarms can be **deleted** by swiping right or left.

Screen 2 - Alarm Detail



This is the **Alarm Detail** screen.
It complies with the Material design principles but does not use a **coordinator layout scrolling** effect, the big tittle has a fixed size and it shows the main **On-Off** switch and a summary for the next planned date for the current alarm, and an alarm **delete** button.
Then it allows to set up and **alarm title** and **time**.
The **repetition** selection box allows to choose from:

- **None** (default): shows a date selection field, where the week day circles are in the current image, and show *today* or *tomorrow* depending on the current selected time, or allows to choose a calendar date.
- **Week days**: the week day circles appear, as shown in the image, and allows to select all or some of them.
- **daily, weekly, monthly, annual:** also shows a date selection field, where the date selection is now mandatory.

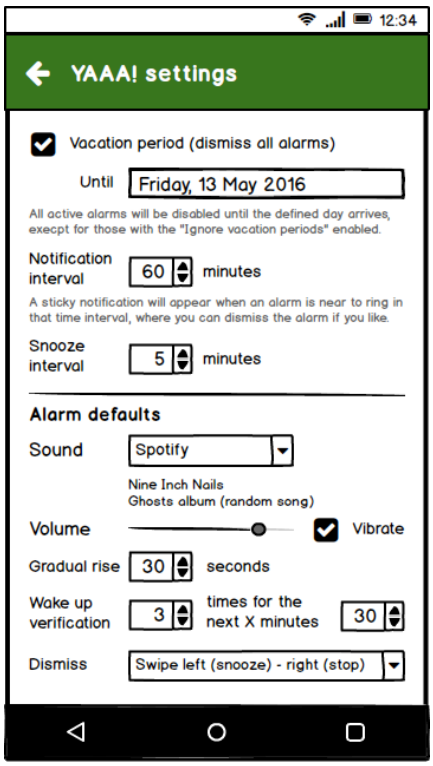
The **sound** selection box allows to choose **default**, or a specific system alarm sound or ringtone, or a locally

hosted mp3 sound file, album or artist folder, as well as a streaming service like Spotify, Last-fm, Shoutcast radios or Google Play Music, to use as the ringtone.
The **volume** level selector allows to choose **default**, or a maximum volume level and the option to **Vibrate**.
The **gradual rise** time**,** in seconds, defines the time interval for the alarm tone to reach the maximum defined volume level. It allows to choose **default** to get the default gradual rise time.
The **wake up verification** option allows to check if the user is really awake by ringing the alarm again, even if it has been stopped, a determined number of times in the next specified interval of minutes. This option overrides the possible snooze selection that the user may choose at any time, it allows to choose **default**, or specific alarm **values**.
The **dismiss** selection box allows to choose **default** or the different alarm dismiss screens:

- **Default** mode: swipe left to Snooze, or Swipe right to Stop.
- **Shake** mode: phone shake or physical button push to Snooze, or Push a screen button (in a random position) to Stop.
- **Conscious** mode: no snooze option and a complex “connect the dots” or some small quiz to Stop, to verify that the user has a correct level of consciousness.

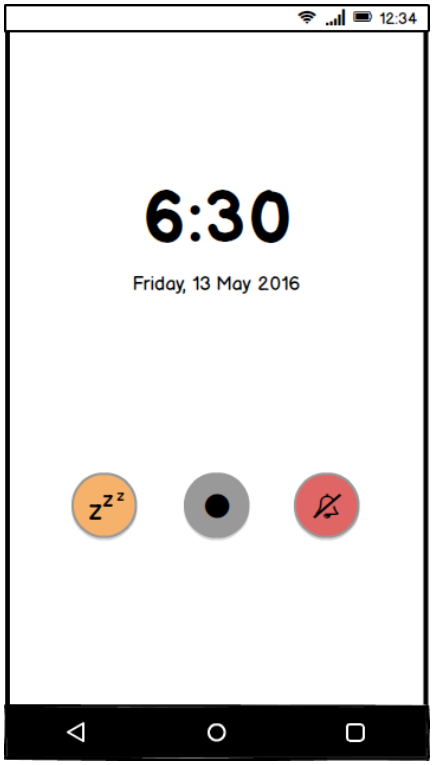
The **delete alarm** option allows to set up an auto-delete alarm time. If the alarm has no repetition, the **when done** check will be enabled, so the alarm will delete itself when its time has arrived, or a finalization date may be set, so the alarm will be deleted the next day after that selected date (usually for repeated alarms).
The final check box **Ignore vacation periods**, unchecked by default, will tell if this alarm will work even if the **Vacation period** setting is active.
The options from **Sound** to **Dismiss** will be grayed out if the **default** option is selected, which means use the default defined in global application settings, as you can see in the **Wake up verification** option in the mockup screen.

Screen 3 - Settings



This is the application’s **Settings** screen.
The **Vacation period** check will allow to disable all active alarms until a date is reached, where they will be enabled again. In that period, all alarm **On-Off** switches are grayed out disallowing any interaction. Except for all alarms with the check **Ignore vacations periods** enabled, that will ignore this setting and work as usual.
The **Notification interval** field defines the time interval where a sticky notification will show up when an alarm event is near to ring. The sticky notification will allow to dismiss that particular alarm occurrence (see screen 5).
The **Snooze interval** field tells the alarm screen the minutes to snooze the alarm if the user chose to snooze. This snooze interval may be overridden if the current alarm has a wakeup verification interval defined.
The next section is for the **Alarm defaults**. These are the same options as for the detail alarm screen but define the default values when they are not specified for a determined alarm.

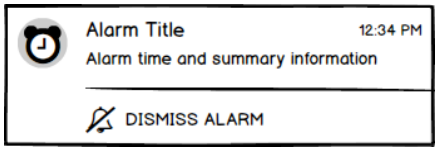
Screen 4 - Alarm



This is the **Alarm** screen.
Will show up at any time when an alarm is activated, and will wait for the user to respond to the alarm event depending on the selected dismiss alarm mode:

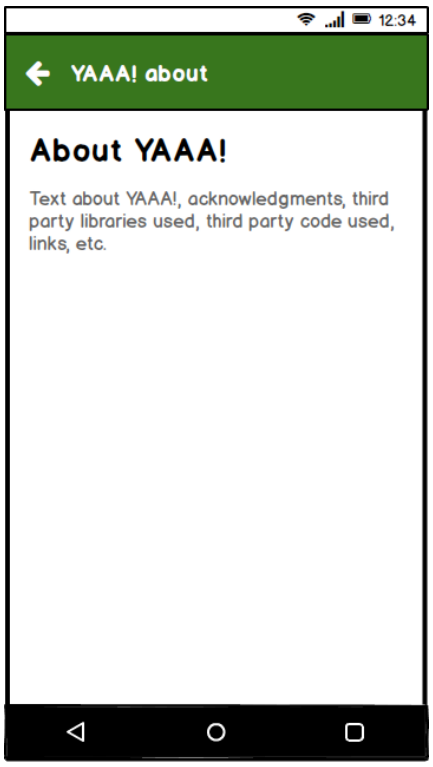
- The **default** mode, where a swipe to left will snooze and a swipe to right will stop the alarm.
- The **shake** mode, where a phone shake or physical button press will snooze the alarm, and a red button press will stop it. The red button will randomly appear on the screen.
- The **conscious** mode, where a small quiz is presented on the screen and the alarm will not stop until resolved.

Screen 5 - Notification



This is a sticky **notification** example. Will show up when an alarm is near to ring as defined in the **Notification interval** global setting. The **Dismiss alarm** button allows the user to dismiss this alarm occurrence.

Screen 6 - About



This is the **About** screen.
Will show a small text about the application, acknowledgements, third party libraries and code used, and links to my personal web, Github profile, Udacity, etc.

This is the initially defined interface and functionalities, but I reserve the right to modify, reduce or enhance any part of this application project.

Key Considerations

Data persistence

The application will use SQLite database to store all the defined alarms, so a content provider will need to be defined, and the simple Shared preferences system to store global application settings and cache data.
Optionally, store all application settings in the Google user's account by using the standard Android Backup Service.

UX interface

The application interface is quite simple and self-explanatory by looking at the mockup screens. The only thing to notice is the difference between phones and tablets, where the Main application (screen 1) and the Alarm Detail (screen 2) screens will appear in a Master-Detail design pattern.

Libraries

There are no specific libraries planned to use, apart from the standard Android libraries like: appcompat-v7, support-v4, support-v13, design, palette, recyclerview, cardview, etc.
But depending on the finally implemented streaming services, it may require to use libraries such as:

- **Volley** (ok-http, okhttp-urlconnection) for artist and album online image caching.
- **Retrofit**, **Spotify-lib**: if the Spotify stream connection is supported.
- **Last-fm-lib**: if the last-fm stream connection is supported.
- **Gmusic.api**: unofficial Android/Java library for Google Play Music if that stream connection is supported.
- **Audiostream-metadata-retriever**: free open library to retrieve metadata information from any streaming source.

Some of these library may have additional dependencies.

Required Tasks

Task 1: Project Setup

Setup an Android Studio project with the usual Android dependencies for a Material design compliant application and with **Jelly Bean (API 16)** as the minimum supported system.

Task 2: Content provider

Define and implement the **alarm database**, its content provider, contract and helper.
Implement all the functionality necessary to get requests sorted by time and filtered by state, and the functionality to get or delete and alarm.
Create a **Test** unit to test its functionality.

Task 3: System service and Screen 5 - Notificaiton

Define and implement the **system service** that will be responsible for checking when an alarm will occur, create the alarm sticky notification and to show the Alarm screen when the time arrives.
Initially create a **Test** work mode using notifications to test its functionality.

Task 4: Screen 2 - Activity holder and Fragment

Define and implement the fragment that will hold the **Alarm Detail** functionality and the activity that will hold that fragment for the phone version.
The interface part that will hold the **Sound**, **Volume**, **Wake up verification** and the **Dismiss** settings will be implemented in another **small fragment**, because it will also be used in the Global settings screen.
This is the largest task as it will also require to implement all necessary custom dialogs to modify all the screen settings.

Task 5: Screen 1 - Activity holders and Fragment

Create and implement the fragment that will hold the alarm **Main** screen and the activities that will hold that fragment for the phone and the tablet versions.
Add the functionality to use the Detail fragment among the Main one for the tablet version.

Task 6: Screen 3 - Activity

Create and implement the **Settings** screen, using the small fragment created in the Task 4. This activity will have the Dialog mode modifier for wide screens (tablets).

Link that screen in the Main activity by using the standard Android title menu options.

Task 7: Screen 4 - Activity

Create and implement the **Alarm** screen with the 3 dismiss modes defined. Add the appropriate activity settings in the Android manifest to allow the activity to unlock the phone.
Modify the application service to start this activity.

Task 8: Screen 6 - Activity

Create and implement the **About** screen activity menu option and add all acknowledgements, third party libraries and code used. Create link objects to necessary webs and profiles.

Task 9: Final steps and testing

Tie everything up, test the application's overall functionality and publish the project!
Usually not a short step, unlike that short description may suggest ☺