Sleep Cycle Alarm Clock for Beddit
Testing documentation

1. Known issues

Id: 001
What: The alarm dialog will pop up on top of a call. The last night sleep info activity will also pop in front of the call.
When: If alarm sets of while user is on the phone
Comments: Might consider this as an acceptable behaviour.

Id: 002
What: Android 2.3.3 emulator runs out of memory:
ERROR/AndroidRuntime(453): FATAL EXCEPTION: main
Caused by: java.lang.OutOfMemoryError: bitmap size exceeds VM budget
When: Application is started for the second time
Comments: Investigate memory leaks? The bug has appeared and then disappeared with no apparent reason.

Id: 003
What: Alarm dialog will open keylock, but when it's dismissed, keylock is activated again. If last night sleep info should display, the screen will be open under the keylock.
How to reproduce: Set "show last night sleep info", set an alarm, press power button once and wait for the alarm to set off. Press dismiss.
Comments: Minor issue at best. This is probably how it should work.

2. End-user acceptance tests

2.1 Authorization:

Scenario 1: User should not be able use the program without logging in

- **Given** there is no user data
- **When** user opens the program
- **Then** Login screen should be displayed

Scenario 2: User should not be able to use the program without authoring it

- **Given** the is at the "Authorize this program" screen
- **When** user presses "No"
- **Then** a dialog should be displayed and the program should close

Scenario 3: Program should accept correct login and authorization

- **Given** the user is at the login screen
- **When** user inputs his/her login info and authorizes the program
- **Then** the main program should be opened

- **and** correct login username and authorization key should be saved in the preferences

Scenario 4: User should not be able to use the program without a valid token

- **Given** the user has cancelled application's authorization via Beddit's website
- **When** user opens the program
- **Then** token should be checked
- **and** user should be taken to the login screen

Scenario 5: User info should be displayed in the settings

- **Given** the user has logged in to the program
- **When** user opens the settings
- **Then** a correct username should be displayed

Scenario 6: Disconnecting should lead to the login screen

- **Given** the user is at the settings screen
- **When** user disconnects his/her Beddit account
- **Then** the user data should be wiped and the login screen displayed

2.2 Main functionality:

Scenario 1: The alarm should display a notification

- **Given** there is no alarm set before
- **When** user sets alarm
- **Then** "Alarm set" message should be displayed
- **and** a notification should appear on notification bar, displaying the set alarm time
- **and** a log message showing calculated wake up time should be in the log

Scenario 2: Moving the clock handles updates alarm

- **Given** alarm is set
- **When** user moves clock handles or interval slider
- **Then** "Alarm updated" message should be displayed
- **and** a notification on the notification bar should display the correct, updated time
- **and** a log message showing calculated wake up time should be in the log

Scenario 3: Removing alarm will remove notification

- **Given** alarm is set
- **When** user presses "Unset alarm"
- **Then** "Alarm removed" message should be displayed
- **and** the notification on the notification bar should disappear

Scenario 4: Alarm should go off at the correct time

- **Given** alarm is set without an interval
- **When** the set alarm time comes
- **Then** a dialog should pop up and alarm tone will play
- **and** corresponding log messages should be displayed

Scenario 5: Alarm should wake up during time interval

- **Given** alarm is set with correct time interval
- **When** the set interval begins
- **Then** the application should start polling sleep stage from Beddit server

- **and** corresponding log messages should display
- **and** the alarm should go off if we have desired sleep stage or final wake up time

Scenario 6: User should be able to snooze the alarm
- **Given** alarm goes off
- **When** the user presses “snooze” button
- **Then** the sound should stop and alarm dialog disappear
- **and** new wake up should be scheduled to wake in 5 minutes (in default) without an interval
- **and** correct notification/message should display in notification bar and log

Scenario 7: User should be able to turn off the alarm
- **Given** alarm goes off
- **When** the user presses “dismiss” button
- **Then** the sound should stop and alarm dialog disappear

Scenario 8: Sleep summary should be displayed after alarm is dismissed
- **Given** alarm goes off and the Sleep info summary is checked from the settings
- **When** the user presses “dismiss” button
- **Then** sleep info summary screen should be displayed

Scenario 9: Sleep summary should display a dialog if connection failed
- **Given** sleep info summary is opening
- **When** connection fails or incorrect data has been fetched
- **Then** an error dialog should be displayed

Scenario 10: Sleep summary should not be displayed if setting is not checked
- **Given** alarm goes off “Show summary of sleep data on wake up” is not checked
- **When** the user presses dismiss
- **Then** sleep summary screen should not open

Scenario 11: Sleep summary should not be displayed if alarm is snoozed
- **Given** alarm goes off
- **When** the user presses snooze
- **Then** sleep summary should not open in any case

Scenario 12: Sleep summary should open browser to the user’s sleep diary
- **Given** sleep summary is displayed
- **When** the user presses “Feeling good” or “I’m tired”
- **Then** a browser should be opened to Beddit sleep diary
- **and** diary should display feeling corresponding to the button pressed

2.3 User-interface

Scenario 1: Setting alarm updates buttons
- **Given** there is no alarm set
- **When** the user presses “set alarm” button
- **Then** “set alarm” button should be disabled
- **and** “unset alarm” button should be enabled
- **and** buttons updated message should be logged

Scenario 2: Unsetting alarm updates buttons
- **Given** there is alarm set

- **When** the user presses “unset alarm” button
- **Then** “unset alarm” button should be disabled
- **and** “set alarm” button should be enabled
- **and** buttons updated message should be logged

Scenario 3: Moving clock handles updates the clock
- **Given** the user is at the main screen
- **When** the user moves clock handles by touch
- **Then** the clock handles should move
- **and** the time display above the analog clock should change accordingly

Scenario 4: Touching the slider should update interval
- **Given** the user is at the main screen
- **When** the user touches slider
- **Then** the interval should be displayed around the analog clock
- **and** the time display should NOT change

Scenario 5: Returning to the program with alarm set displays correct state
- **Given** alarm is set and user is not in the program
- **When** the user return to the program by any means (presses back, opens the program again, etc.)
- **Then** the buttons should be set accordingly
- **and** the clock should display the set alarm time

Scenario 6: Returning the program with alarm not set will display old alarm
- **Given** alarm is not set and user is not in the program
- **When** the user returns to the program by any means (presses back, opens the program again, etc.)
- **Then** the buttons should be set accordingly
- **and** the clock should display the time alarm was last set

Scenario 7: Sleep summary should not display buttons if opened via menu
- **Given** the menu is open
- **When** the user presses Sleep info
- **Then** sleep summary screen should be opened
- **and** sleep summary should not display buttons

2.4 Menu behaviour

Scenario 1: Menu has three buttons
- **Given** user is logged in
- **When** the user presses menu button
- **Then** menu should be opened displaying buttons “Settings”, “Help” and “Sleep Info”

Scenario 2: Help displays help text
- **Given** menu button has been pressed in main activity
- **When** the user presses help
- **Then** help text should be displayed

Scenario 3: Sleep info should open last night summary
- **Given** menu button has been pressed in main activity
- **When** the user presses Sleep info

- **Then** sleep summary should be opened

Scenario 4: Setting button will display settings
- **Given** menu button has been pressed in main activity
- **When** the user presses settings
- **Then** settings should be opened

Scenario 5: User should be able to set snooze length
- **Given** settings is opened
- **When** the user presses Snooze time length
- **Then** user should be able to choose snooze time length
- **and** the snooze length should be the length chosen

Scenario 6: User should be able to set in which sleep stage the app will wake up
- **Given** settings is opened
- **When** the user presses Sleep stage to wake up
- **Then** user should be able to choose sleep stage
- **and** the alarm should take this setting into account

Scenario 7: Advanced settings displayed
- **Given** settings is opened
- **When** the user presses Advanced settings
- **Then** advanced settings should be displayed

Scenario 8: Colour theme
- **Given** advanced settings is opened
- **When** the user presses “Colour theme”
- **Then** user should be able to change between “light” and “dark”
- **and** the corresponding theme should be in use in the program

2.5 Localization

Scenario 1: Default language is english
- **Given** the phone’s language is not supported by this program
- **When** the application is used
- **Then** all texts in the application should be in english

Scenario 2: Supported language is displayed
- **Given** the phone’s language is a language supported by this application
- **When** the application is used
- **Then** all texts in the application should be in the corresponding language

Scenario 3: AM/PM time displays correctly
- **Given** the phone’s time is in AM/PM display mode
- **When** the application is used
- **Then** clock display time and notification times should be displayed in AM/PM time

Scenario 4: 24h time displays correctly
- **Given** the phone’s time is in 24h time mode
- **When** the application is used
- **Then** clock display time and notification times should be displayed in 24h time

2.6 Special cases

Scenario 1: The volume is off if user is on the phone

- **Given** alarm is set and user is on the phone
- **When** the alarm goes off
- **Then** the alarm tone should be muted
- **and** the phone should not vibrate

Scenario 2: The volume is low if the alarm goes off at the same time someone is calling

- **Given** an alarm is set and the phone is ringing
- **When** the alarm goes off
- **Then** the alarm tone should be quiet
- **and** the phone should not vibrate

Scenario 3: Disconnecting should delete set alarm

- **Given** an alarm is set
- **When** the user disconnects his account and confirms the deletion
- **Then** the alarm should be removed

Scenario 4: Canceling on disconnect should not delete set alarm

- **Given** alarm is set
- **When** the user presses disconnect, and then “no”
- **Then** the alarm should not be removed

Scenario 5: User should be able to use the app without internet connection

- **Given** user has logged in before but has no internet connection
- **When** the user opens the program
- **Then** program should open without problems

Scenario 6: Sleep info shows dialogue without internet

- **Given** the user has no internet connection
- **When** the user presses Sleep info or alarm dialog should show sleep summary
- **Then** sleep info should display “Could not connect” dialog

Scenario 7: Alarm goes off without internet connection

- **Given** the user has no internet connection and alarm is set
- **When** the set interval begins
- **Then** application should try to connect Beddit for sleep information
- **and** finally wake up at the last wake up time

Scenario 8: Default time is 8:00AM

- **Given** the user has not used the program before
- **When** main screen is opened
- **Then** clock handles and time should be set to 8:00 AM

3. Testing

At the beginning of the project, the team’s goal was to make automatic unit tests and behaviour tests for as much as possible. The team however met unforeseen problems testing Android projects with IntelliJ IDEA. Too many work hours were wasted just to figure out how to get the

tests work on multiple machines, and diagnose why the tests didn't run correctly.

The automatic tests became a burden. If a test didn't pass, it was practically always due to the new features or complete refactoring. Testing the real logic behind alarm clock was frustratingly hard due to the fact that it's (for now) impossible to give correct permissions to change the Android system time for project tests. UI testing with custom components would need a lot of code and time to simulate correct touch events for the custom GUI components. The behaviour tests would not always run as intended if there was still old user or alarm data saved on the phone from previous test runs. Automatic tests were also not able to track notifications on notification bar.

In the end, automatic tests exist only for the main logic class behind alarms, AlarmService. Even with that, the JUnit tests are only testing that correct services are called - the application is presumably doing what's it supposed to do. The automatic testing was abandoned since it would have taken more time than doing the testing manually and that extra time would have been cut off from programming and bug fixing. It was a lot faster to test all the relevant parts by hand. By manually testing the program the team was also able to find a huge amount of bugs which would have not been detected by automatic tests.

3.1 Manual and exploratory testing

To compensate the lack of automatic tests, the team has done active manual testing every day. The manual testing was eased by huge amount of log messages. Every time something was changed, the relevant parts of the program and probable victims of the changes were tested manually. Almost all the members were also using different Android emulator versions (2.2, 2.3.3 and 4.0.3) and real Android phones to check UI and parts of the program with different Android phones. The main thing that was fixed by this were scaling the custom UI parts for different screen configurations and Android versions. Connection problems were fixed for 2.2, where the HTTP-connection behaved differently than newer Android versions. With this method the team was also able to learn about features that work differently with emulators and phones. One of these bugs was found during the final customer demo, where we were able to test the application on a phone which had a keyboard for the first time. Unfortunately more bugs are likely to surface in this manner as the number of phones and different hardware configurations to test on during development was limited.

Manual testing was done by testing simple most basic scenarios, most of which were also documented under end-user acceptance tests in the section 2 of this document. At the same time as the team was testing the program's functionality, attention was paid to the UI and localization issues.

A lot of time was used on reviewing each others code and refactoring. Reviewing other team members' code was used effectively - all the members looked almost all the code at some point of the development. It also helped with finding several fatal bugs, refactoring to a more simpler code - understandable, more consistent code style and better (though not perfect) variable and class names.

Almost all the time someone was also exploring the program as a whole and trying to find flaws. After the main features became more stable and didn't change anymore, the main focus moved to special cases and bugs that might rise from the Android environment's special cases. This included for example phone's sleep mode and behaviour of the back and main buttons at

different parts of the application, and abusing the program by for example spamming buttons multiple times.