**myCalendar App User Manual**

**(insert app screenshots here)**

**Authors:**

Group 7

Alejandro Arredondo

Izuchukwu Elechi

Jared Franze

Taber Hust

Andrei Stringfellow

**App Description:**

myCalendar works like most calendar applications and gives the user the following features:

Calendar Views

* Monthly View: shows all the days in a month. A small dot will appear on any day with a scheduled event.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Weekly and Agenda View are combined, the top third of the interface shows the

\* current week, while the bottom two thirds contains a scrollable agenda view

\* consisting of all events from the current day of the week forward.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

* Weekly/Agenda View: Shows current week with highlighted current day. As in monthly view, a small dot will indicate the presence of an event. Below current week is the agenda view which shows all future events in a scrollable view starting from the current day/first upcoming event.

Calendar Events

* User can create an event with starting and ending time; app also checks for time conflicts when creating events
* User can create daily, weekly, and yearly periodical events
* event categories //Not implemented yet
* Color marking for different category of events //Not implemented yet
* User can edit & delete events
* Event reminders can be set. These implement the default alarm method set on the user’s device.

The app also includes unique features to distinguish it from other calendar applications:

Audio Events

* Add an audio recording to calendar as an event
* Audio recording reminders //Could be difficult to implement
* Schedule the frequency of reminders
* Add/delete audio events

Other Features

* Holiday & weekend color coding
* This Day in History

Calendar Share Features

* Events from all pre existing calendars are added to the calendar application. Also, any changes made within the application will persist to other calendar applications.

**Source File Description:**

**Agenda.java:**

This is the main activity for the Calendar. Upon opening, this activity is created and the user is presented with the week view/agenda view combination screen. Loads all pre existing events at startup into the application database.

**CalDBCommun.java:**

This is the core of Event database operations. Contains the active lists of Calendar and Event objects present in the system. Has capabilities to call methods which implement CalendarAdder, EventAdder.java, EventRemover.java, and EventUpdater.java. This class checks for conflicts when adding events, and has the ability to delete all events present in a system.

**CalendarAdder.java:**

This class is responsible for adding a Calendar object to the calendar list stored in CalDBCommun.

**Date.java:**

Creates a Date object to be used by the Event class for easy access to commonly used variables such as YEAR, MONTH, DAY, HOURS, MINUTES, SECONDS.

**Event.java:**

Houses Event objects which contain all necessary attributes of our events for use in our calendar application. Contains many set() and get() functions to access and change Event data.

**EventAdder.java:**

The constructor of this class takes an Event object and adds it to the Event list housed in CalDBCommun. These values are also added to the CalendarContract database.

**EventIcon.java:**

Contains assorted icon information for use in User Interface.

**EventRemover.java:**

This class houses the overloaded method, deleteEvent. The first instance of deleteEvent takes an Event ID and removes the Event associated with the ID from the event list and the CalendarContract database. The second instance takes in an actual Event object and removes that object.

**EventUpdater.java:**

This class contains the method updateEvent which receives two event objects, the old Event and the new Event. Maintaining ID, all new Event attributes are set in place of the old Event data.

**ListInit.java:**

This class is in charge of initializing the Calendar and Event lists implementing CalendarContract.

**MonthView.java:**

This is a separate activity that allows the user to navigate to month view within the application. Here, the user is able to view any past of future month in its entirety. MonthView implements the calendarView and uses a datePicker dialog to select a date.

**myCalendar.java:**

This class is in charge of creating the Calendar object that our application will be using.

**ThisDayInHistory.java:**

Obtains an RSS feed from History.com containing continuously updated ‘this day in history’ information. Titles and a short description are saved as attributes to the Event object.

**VoiceMemo.java:**

This is a separate activity which allows the user to record a voice memo and attach it to a specific event for later playback. The user has the option to playback before saving and can record as many times as necessary until he/she is satisfied with the recording.

**How to Run:**

1)

As with any android application, once installed onto the system, you only need to press on the application icon.

2)

User creates a new event in Agenda View by pressing the ‘plus sign’ icon on the upper right corner of the page. Here, the user inputs Event attributes.

**Adding Voice Memos:**

Voice memos can only be added to an already created event. Once created, select the event within agenda view and press on the microphone icon. You will then be prompted to record your memo. To record over a preexisting memo, press and hold the microphone icon