GitHub Username: harryio

Flubo

Description

Never miss a meeting or a deadline or an appointment or any other task in your busy day and always stay up to date on what's next. Flubo is a task manager app that keeps track of your important tasks and habits and notifies you at the right time so you never miss them making you more efficient and punctual. Plan your day or week or even a year in advance and Flubo will notify you when it's time to do your task.

Intended User

This app can be used by anyone who would like to manage their tasks.

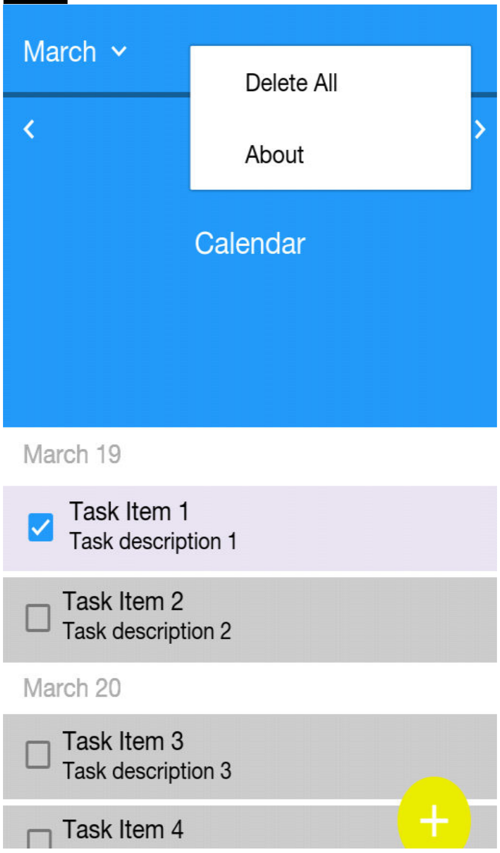
Features

- Create one time tasks
- Create recurring tasks
- Choose timing of your tasks
- Tasks are organized by day, so you can always see what your day looks like
- Show notification when it's time to do your task
- Play sound when alarm triggers
- Edit your already created tasks
- Color code your tasks by choosing from range of colors
- Snooze your alarms

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



March 19

☑ Task Item 1
Task description 1

□ Task Item 2
Task description 2

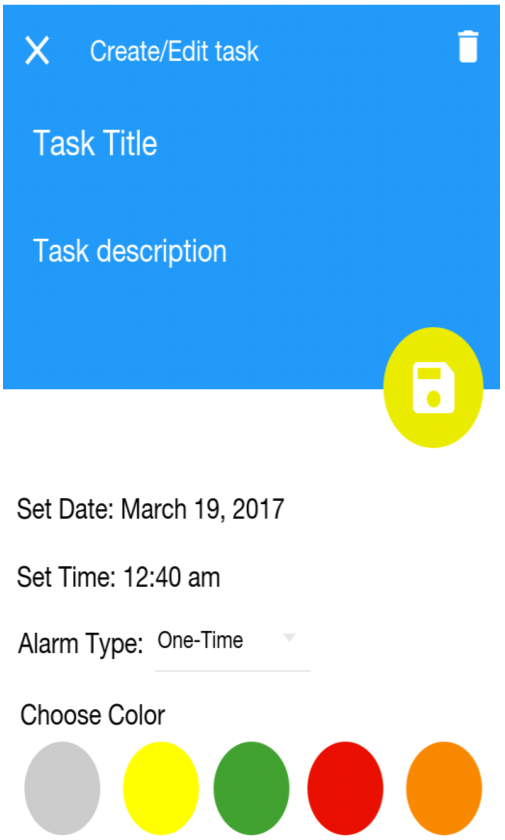
March 20

□ Task Item 3
Task description 3

□ Task Item 4

MainActivity UI

Screen 2



Task Title

Task description

Set Date: March 19, 2017

Set Time: 12:40 am

Alarm Type: One-Time

Choose Color

Create/Edit Activity UI

Add as many screens as you need to portray your app’s UI flow.

Key Considerations

How will your app handle data persistence?

App will store task data in sqlite database and will use external library to handle the database operations

Describe any corner cases in the UX.

If a user is editing an already created task, should the edited fields be automatically updated in the database or should we ask user whether to save the edited fields.

Describe any libraries you'll be using and share your reasoning for including them.

A database library to handle operations on Sqlite
RxJava and RxAndroid for loading data from database in a background thread

Describe how you will implement Google Play Services.

Can't think of any way Google Play Services will be used in the app in the current stage. Maybe in the future.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Choose initial activity as blank and name it as MainActivity
- Use latest support libraries
- Make app support Kitkat and higher
- Search and add suitable database library

Task 2: Implement UI for MainActivity

- Create custom view for Calendar
- Add recycler view to the MainActivity to show list of tasks
- Create UI for task list item
- Add FAB at the bottom right of the screen to allow user to add new task.
- Create toggle for showing/hiding calendar and place it on toolbar
- Add menu option to Delete all of the saved tasks
- Add menu option for About page

Task 3: Add functionality for MainActivity

- Take user to the selected day tasks when user clicks on a day in custom calendar view.
- Show UI flag for dates for which tasks are present in custom calendar view
- If no task is present on selected date, then show user the option to create one for that particular date
- Animate in/out custom calendar view on toggle
- Fetch saved tasks from the database on a background thread
- Write an adapter to show fetched tasks in the RecyclerView on main thread
- Show empty screen if there are no saved tasks

- Allow user to mark a task 'done' manually
- Allow user to delete a task by swiping the coressponding list item to the right
- Show snackbar for a limited period to undo deletion of a task
- Implement OnClickListener task list item
- Take user to edit task screen on clicking the task list item
- Take user to create new task on clicking on FAB in the bottom right of the screen
- Add functionality of "Delete All" menu action
- Show snackbar to allow user to undo deletion of all items
- Take user to the about page on clicking "About" menu action

Task 4: Implement UI for creating new task

- Create new Activity for creating a new task.
- Create Edittext for title of the task
- Create Edittext for description of the task
- Create view/s for setting date for the task
- Create view/s for settign time for the task
- Add option for selecting between one-time and recurring tasks
- Set a system alarm for user selected day and time
- Add color choose to color code the task
- Add option on toolbar to close the activity
- Add Button to save the new task

Task 5: Add functionality for creating new task activity

- Save newly created task to the database when user clicks on "Create" button.
- Check for task data validity before saving it to the database. For e.g Task title cannot be empty, etc
- Finish the activity when the new task is saved to the database.

Task 6: Implement UI for editing a task

- Show task data in a UI similar to create new task UI(Probably in the same acitivity as create new task Activity)
- Make all views editable so that the user can edit them
- Add menu action to delete the task
- Add button to save the updated task

Task 7: Add functionality for editing a task

- Check all fields of a task to determine whether the task data has actually been changed or not.
- If the task data is changed, then save the updated data(or manually ask the user to save it or not) in the database and finish the activity
- If the task data is not changed, then simply finish the activity
- Delete the previous alarm if the user changed the time for the task and set a new one
- Delete the item from the database when user click the "Delete" menu action

Task 8: Notify user

- Notify user through a Notification or alarm sound when an alarm goes off for a task.

Task 8: Create UI for snoozing an alarm