GitHub Username: torbenmoeller

Schmöker - RSS Reader

Description

Schmöker is an app to subscribe to your favorite web feeds. It supports the most common web formats like RSS and Atom. It lets you managed your web feeds easily.

Intended User

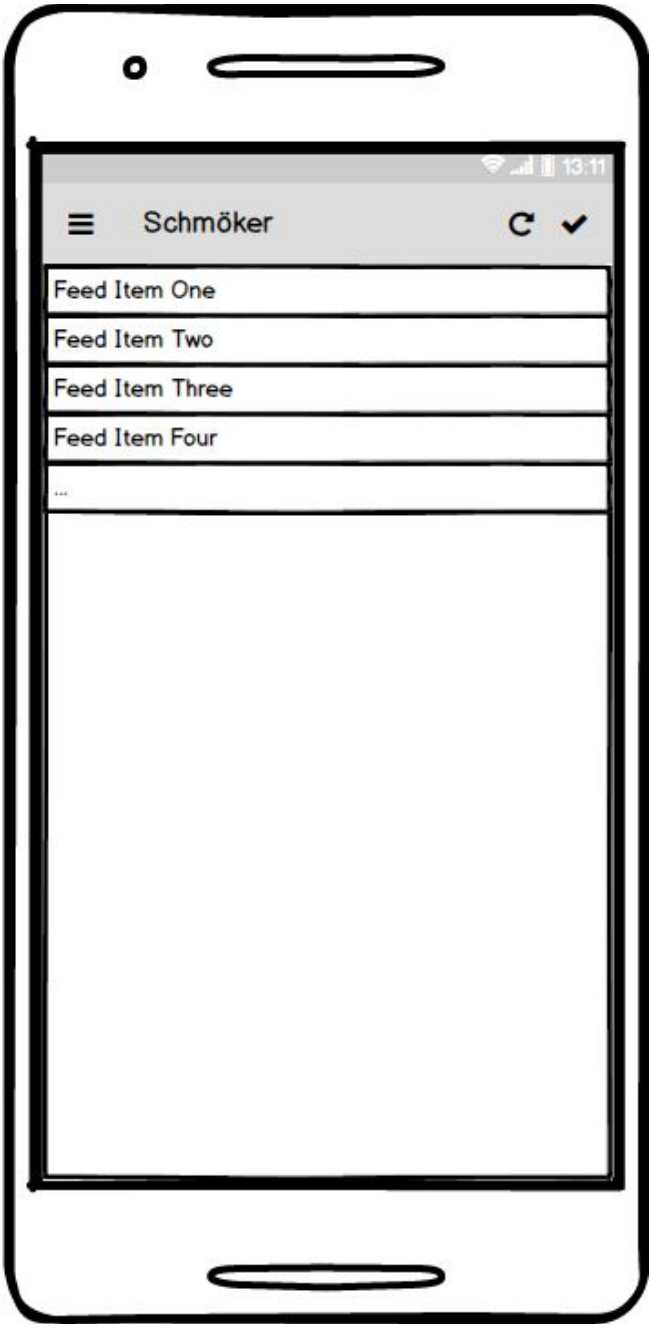
The intended user is everyone who wants to use multiple web feeds in order to stay up to date. A web feed allows content distributors to broadcast updates via a standardized format. For example most newspaper and tech website host web feeds to broadcast news. The most common formats are RSS and Atom feeds. It is similar to facebook or twitter feed, but what content the feed contains is determined by the choices of the user. Users can subscribe to a variety of web feeds and therefore the interests of the inteded users are heterogeneous.

Features

- Support RSS and Atom feeds
- Subscribe to feeds and unsubscribe feeds
- Pulls updates regularly
- Show merged newsfeed from multiple sources
- Get notifications for news
- News can be read offline
- Widget for showing news the home screen

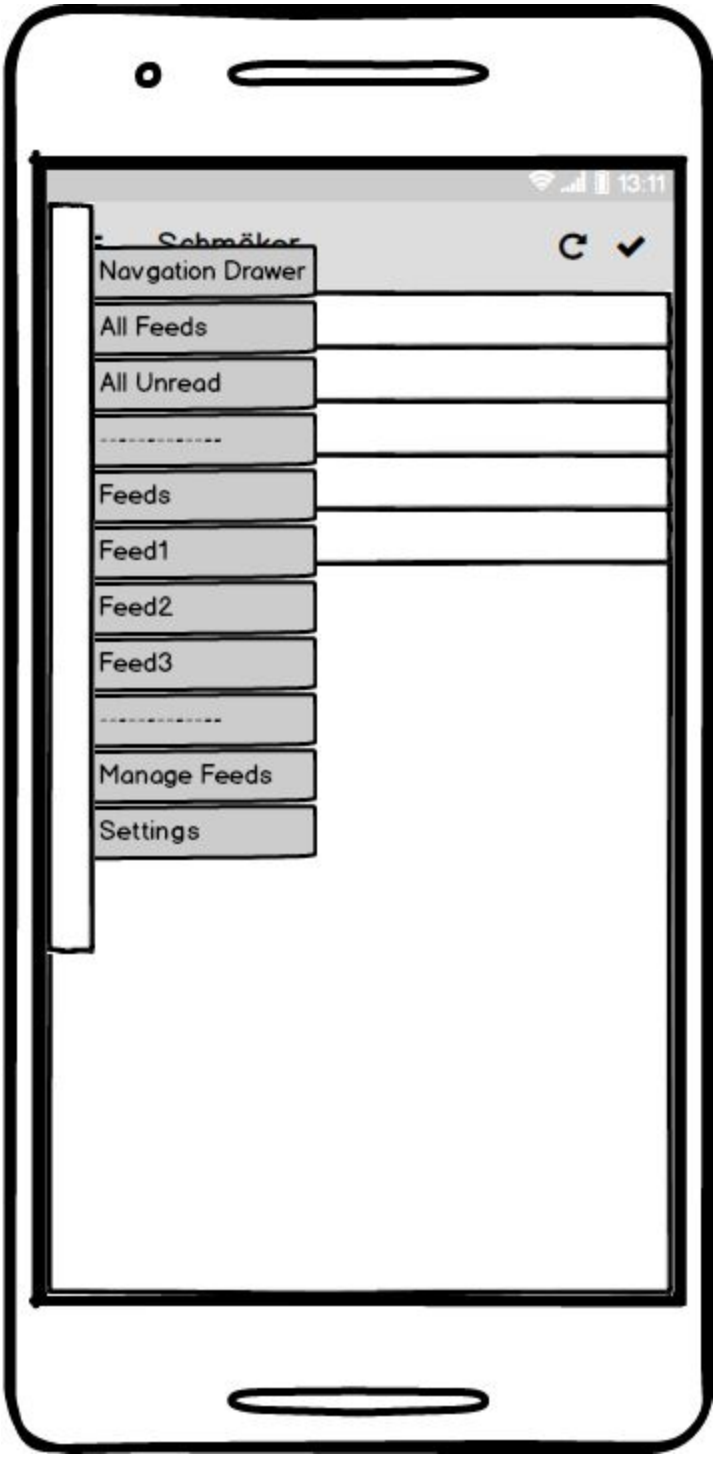
User Interface Mocks

Main Screen



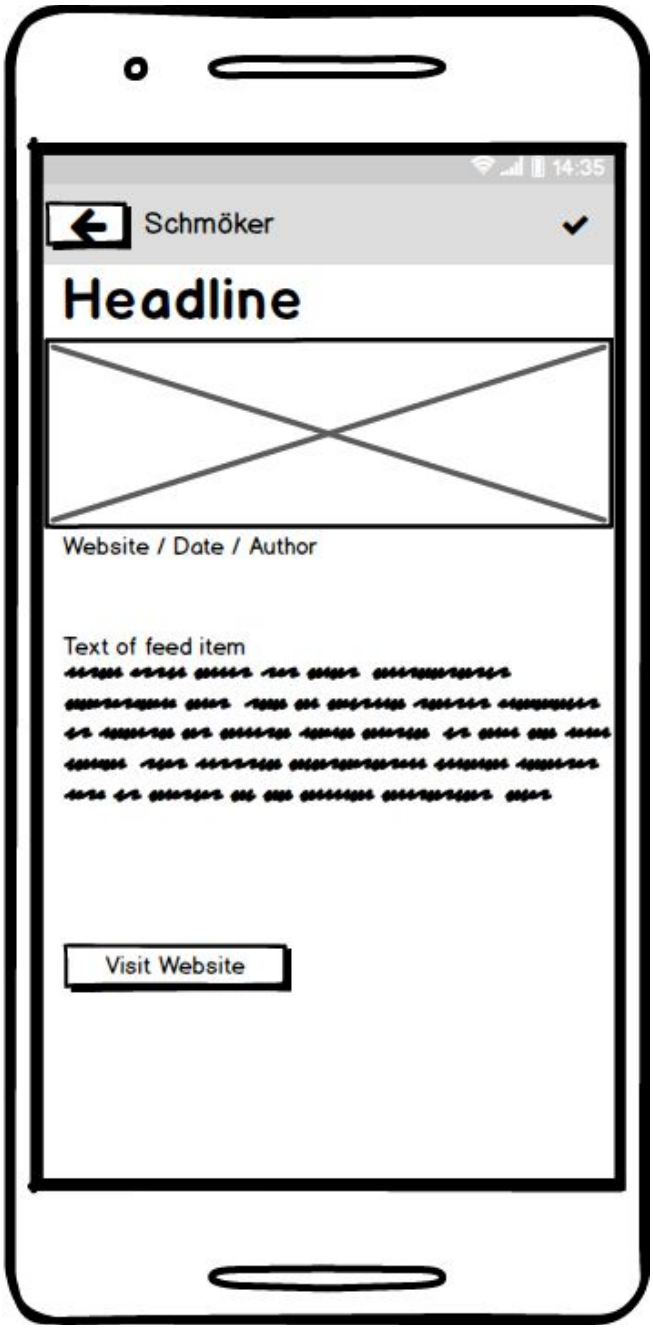
Shows all new articles (read and unread), sorted in the chronological order. A swipe down or the refresh button in the app bar, starts a new pull of articles. The check button in the top right marks all unread articles as read. The button in the top left opens the navigation drawer. A click on the article opens the article screen.

Navigation Drawer



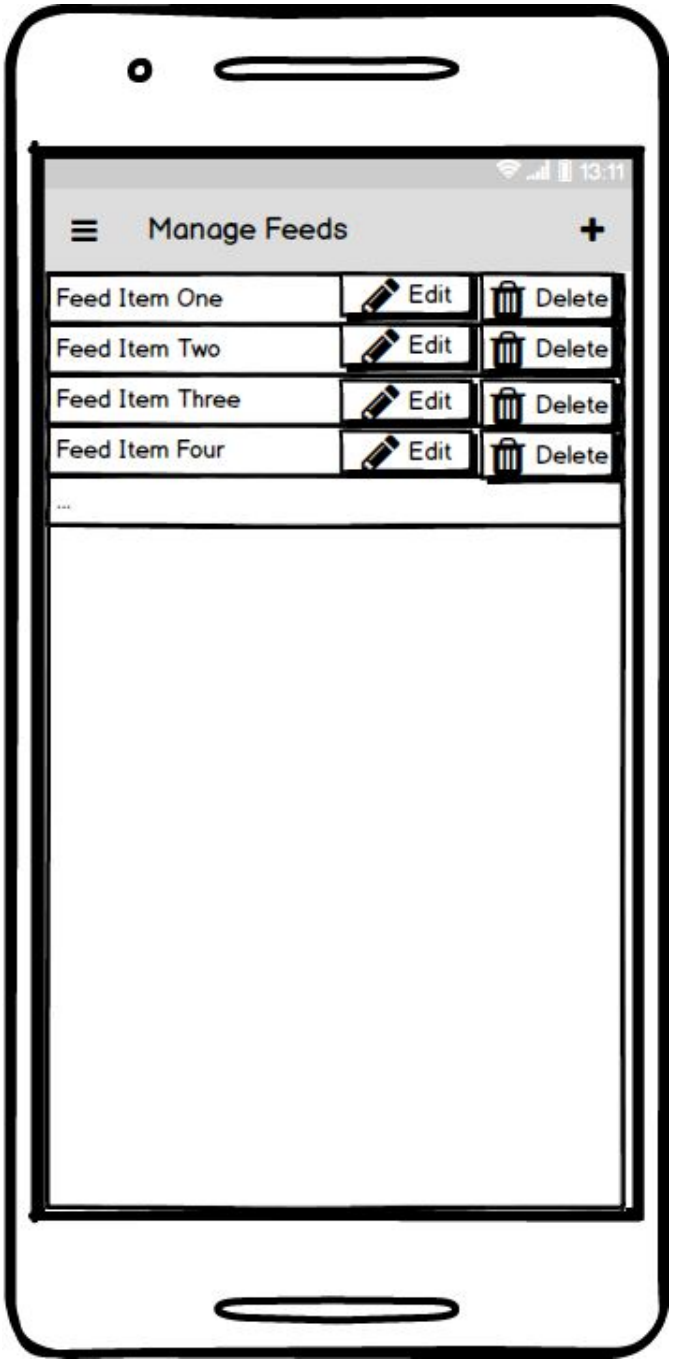
The navigation drawer shows the feeds you can open. “All feeds” shows all feeds merged into one. “All Unread” shows all unread articles from all feeds. Also each feed can be opened and then shows only news of the selected source. “Managed feeds” shows the screen to manage feeds. “Settings” shows the settings screen.

Article Screen



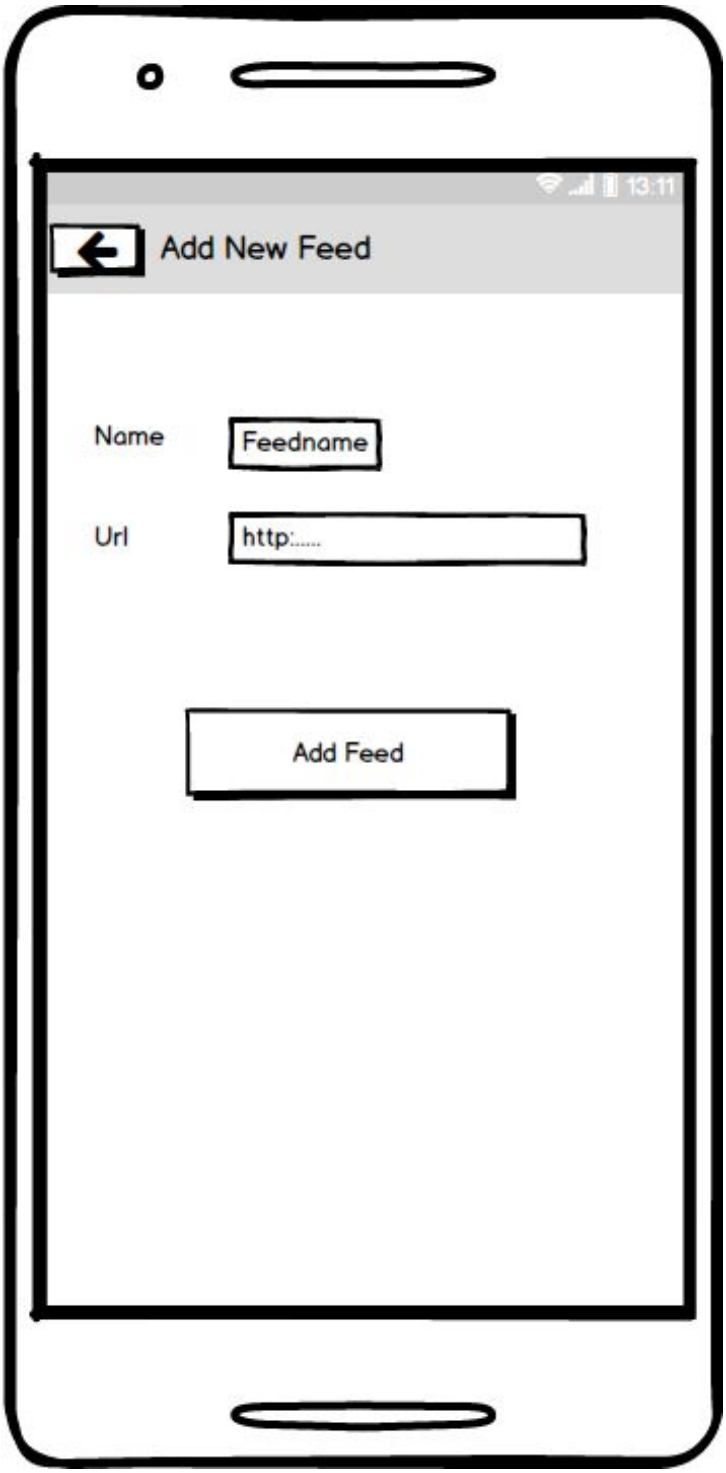
The article screen shows the headline of the news. A picture is shown, if one is available in the feed. Below the image there are the name of the source, date of the article and author (if available). The text from the feed item is shown right in the middle. “Visit Website” opens the browser if there is a link in the feed item available.

Manage Feeds Screen



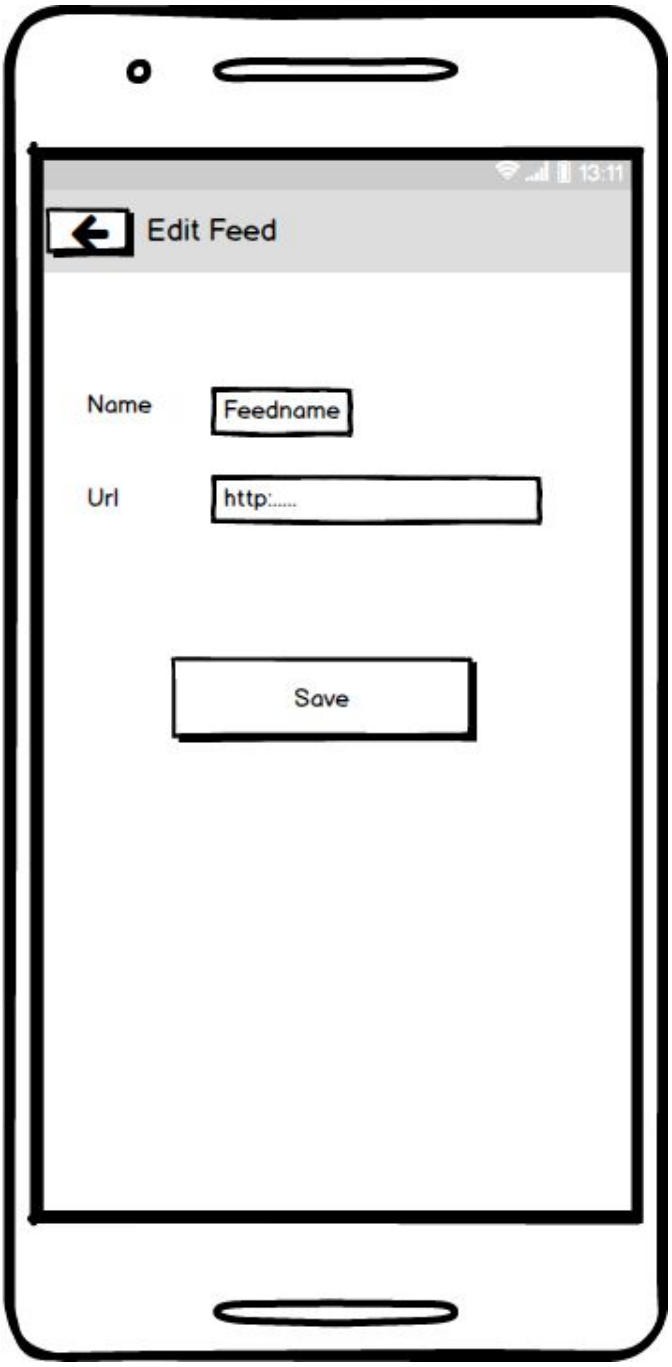
The managed feeds screen allows easy management of all subscribed feeds. The plus button in the app bar opens the add new feed screen. The edit buttons open the feed in the edit feed screen. The delete button opens a confirmation to unsubscribe the feed. After a feed was added, edited or deleted this view is updated.

Add New Feed Screen



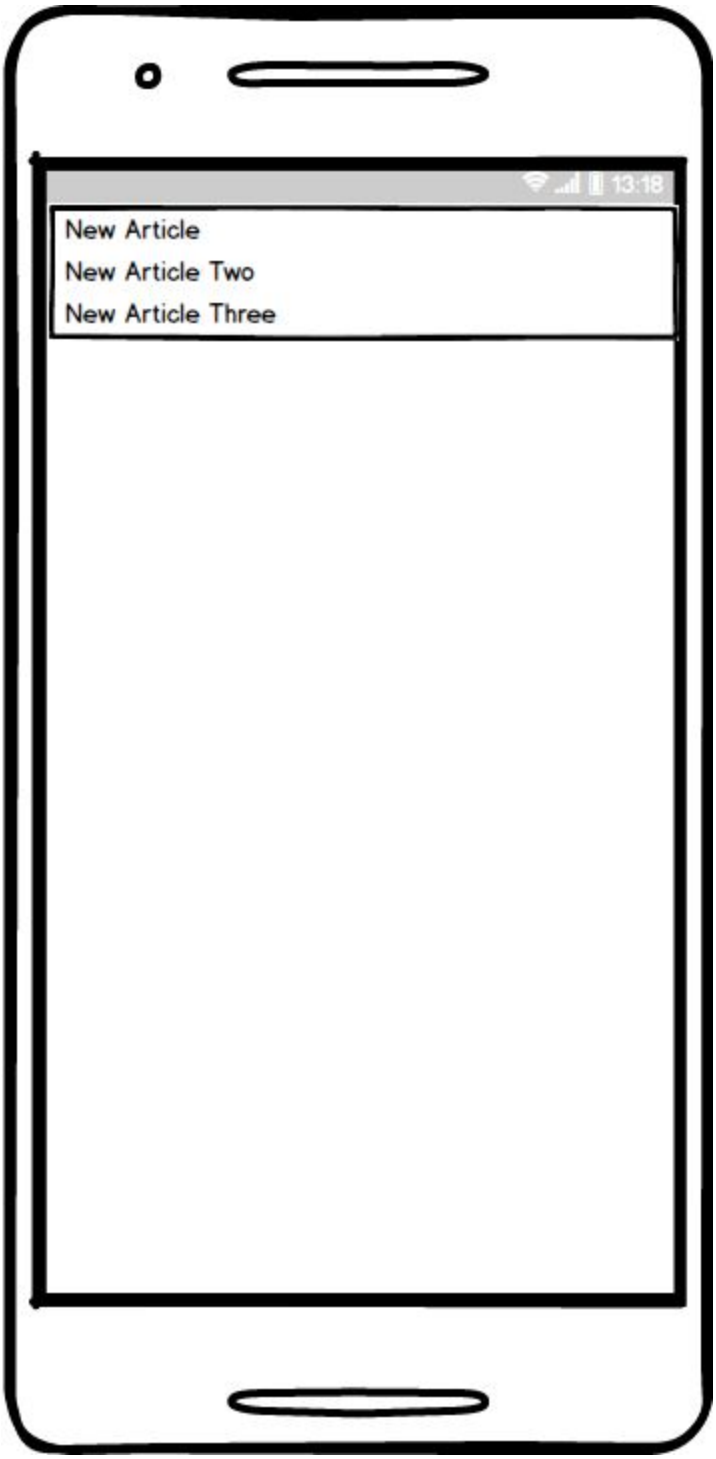
This screen adds a new feed with name and url, if the add feed button is pressed.

Edit Feed Screen



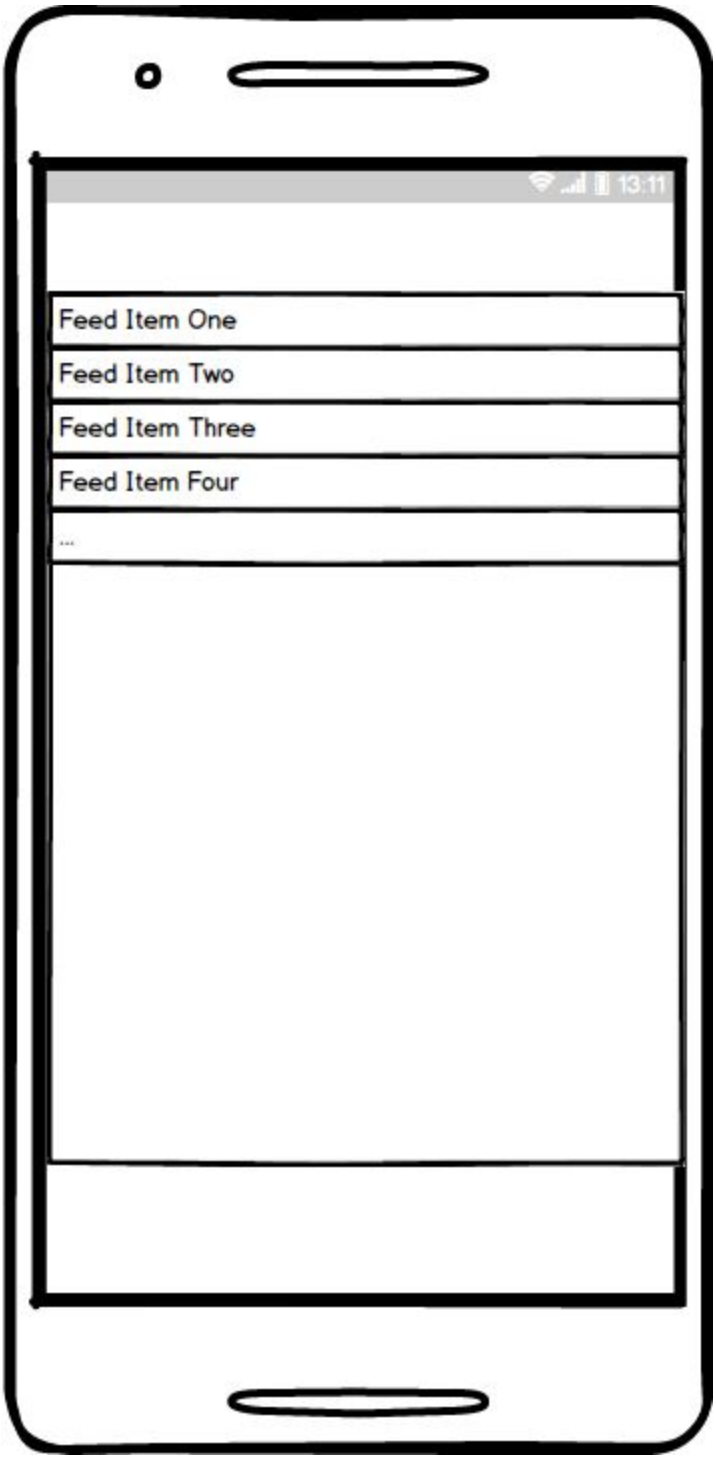
This screen updates a feed with name and url, if the save button is pressed.

Notification Bar



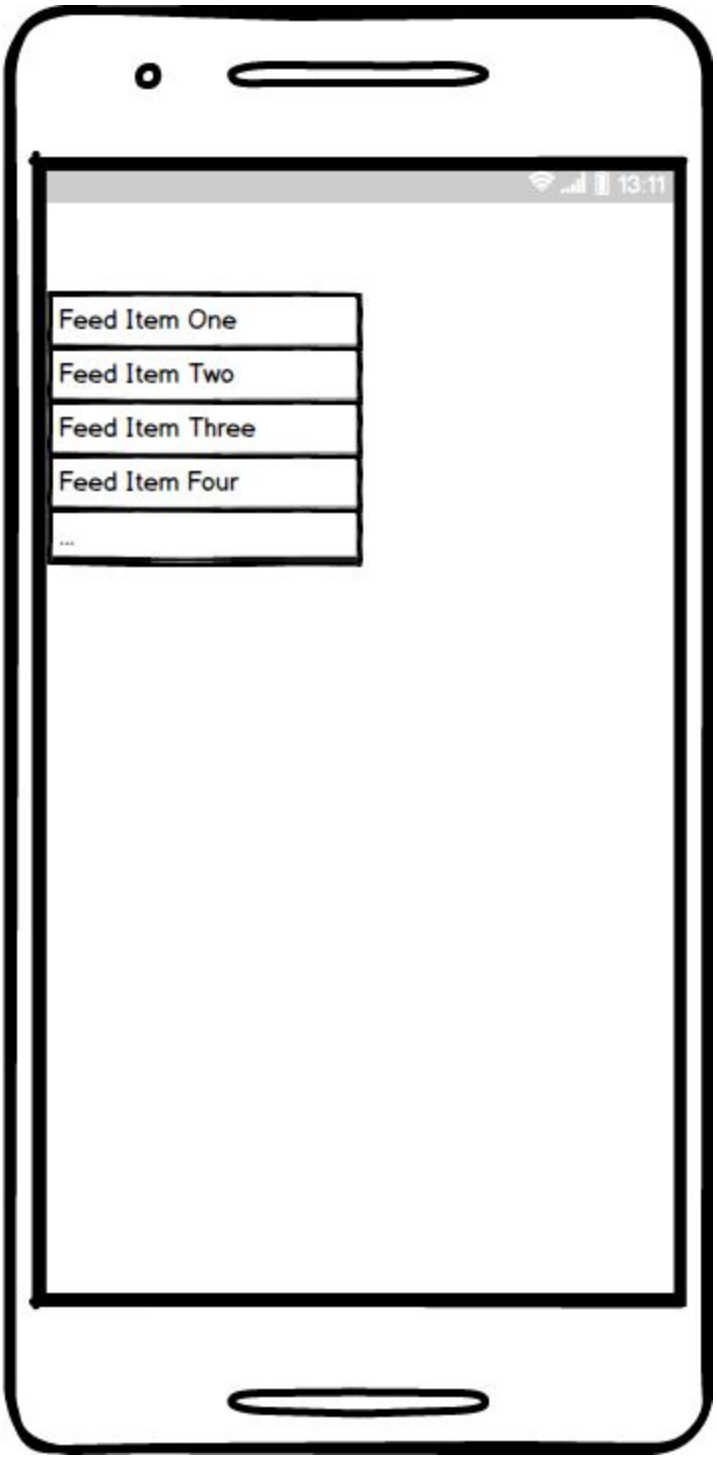
The notification shows current unread articles. A click opens the apps' main screen.

Widget Large



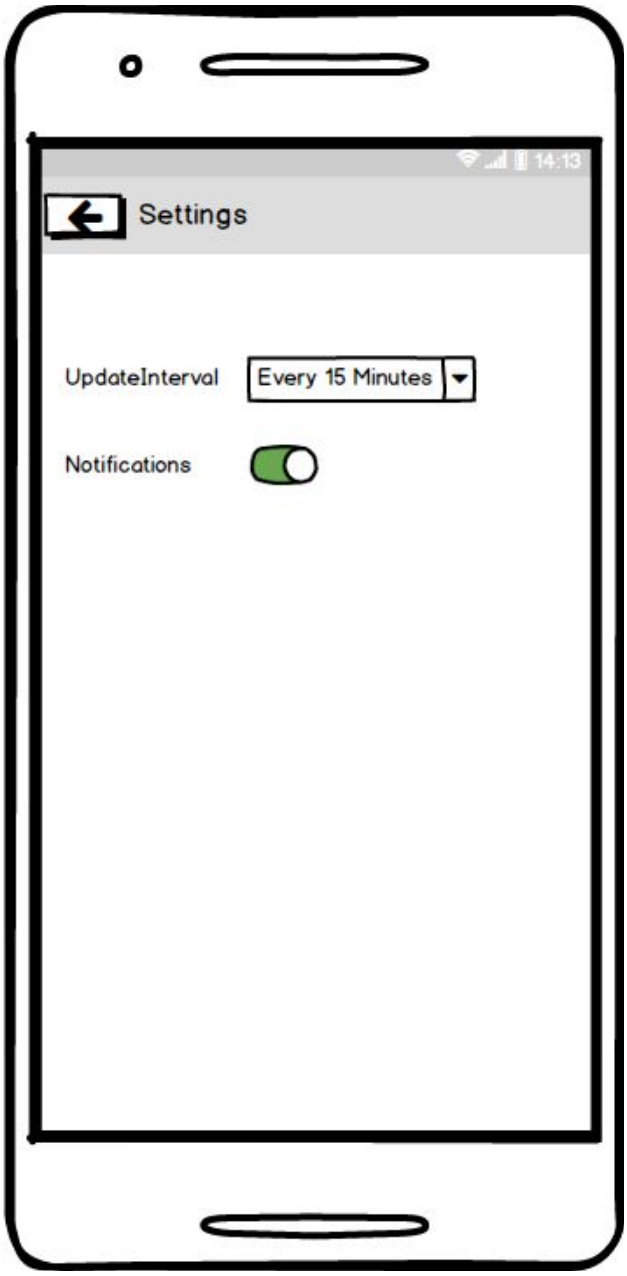
The widget shows the all feeds screen.

Widget small



The widget is also scalable and show fewer articles if smaller.

Settings



The settings screen allows the following settings:
Change the Updateinterval to “Every 15 minutes”,“Every 1 hour”, “Every 2 hours”, “Every 4 hours”, “Daily”
Notifications can be enabled or disabled.

Key Considerations

How will your app handle data persistence?

Every feed subscription is saved in a table. The subscribed feeds are synchronized to a local database, so every article is available offline. The database also contains a table to save if a article was read. Room is used as library.

Describe any edge or corner cases in the UX.

If a feed is not available it is shown on the feed page. Unread news are shown bold font to visualize if there are unread news. Empty or unavailable feeds are shown with an error sign in the navigation drawer. If a manual refresh leads to an exception it is shown via a snackbar.

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

Room, for data persistence
Rome, to pull and parse RSS feeds
Picasso, if a feed contains images.
Espresso, Mockito, JUnit, Hamcrest for testing.
Android Support Libraries if necessary.
And all necessary transitive dependencies.

Describe how you will implement Google Play Services or other external services.

No Google Play Services needed.

Next Steps: Required Tasks

Tasks contain tests if necessary.

Task 1: Project Setup

- Create GitHub Repository
- Add Libraries
- Choose some RSS/Atom feeds for development and testing purposes
- Initial commit

Task 2: Implement Feeder

- Implement a local available feeder for test purposes
- Implement RSS feeder
- Implement Atom feeder
- Implement Feed merging

Task 3: Implement UI To Show Feeder

- Create feeder layout
- Implement adapter

Task 4: Implement Navigation Drawer

- Implement Navigation Drawer, to switch between feeds

Task 5: Implement Feed Management

- Create feed management layout and functionality
- Create add feed layout and functionality
- Create edit feed layout and functionality

Task 6: Implement Notifications

- Implement Notifications for updates

Task 7: Implement Settings

- Implement settings layout and functionality

Task 8: Implement Notifications

- Implement Notifications

Task 9: Implement Widget

- Implement Widget and functionality