User Documentation

The following is a guide to the use of the MemeDoppler app. This guide starts on the assumption that the user has the most recent version of the app open on a phone or emulator of their choice. Details on how to reach that state are found in the Developer Documentation.

Future Final State

This section explains the planned functionality of our app. Below, missing and partially functional features will be listed.

Main Screen

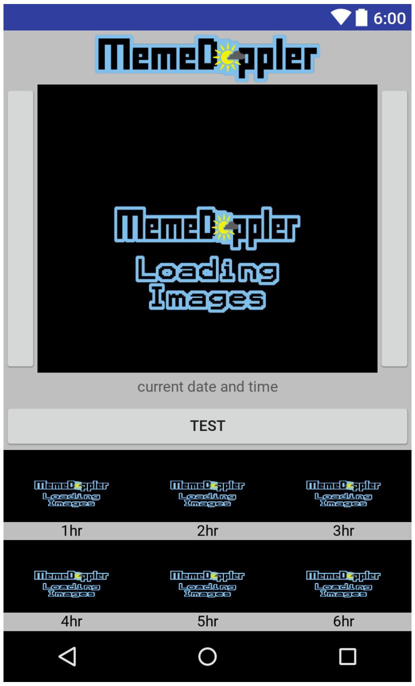
When you first open the app, the first thing you will see is the home page of our app, the main screen. Depending on the speed of your internet connection and device hardware, you may or may not see our loading images in our image panes. The central image represents the current weather at your location. The current location is listed on a button spanning the middle of the screen. Location is explained below. Above this button, the timestamp of the most recent refresh is listed. Below these are the forecast panes. They are in chronological order, from 1 hour in the future to 6 hours in the future left to right, top to bottom. They are labeled as such, which the label being located below its corresponding pane. The buttons to the left and right of the image are used in user feedback, as explained below.

User Feedback

User feedback is activated in one of four ways, two positive and two negative.

Clicking on the tall, thin button to the right of the central image activates positive feedback. Swiping from left to right across the screen also activates positive feedback. On activation of positive feedback, data is sent to the server and a short thank you message pops up, and shortly fades.

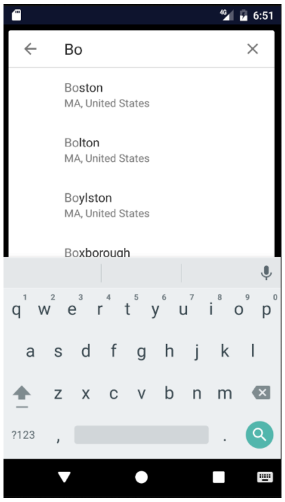
Clicking on the tall, thin button to the left of the central image activates negative feedback. Swiping from right to left across the screen also activates negative feedback. On activation of negative feedback, we transition to a new screen. This offers several related images as preferable options to the current image. Clicking on an image will send data back to the server showing your preference. Clicking on the large button at the bottom of the screen sends data back to the server showing your lack of preference for all offered images. After selecting an option, you are sent back to the main screen and another thank you message is displayed.



Location

Users have the option to select their own location. This is done by clicking the long thin button in the center of the screen, with the name of the current location. This brings us to a new screen, with a search bar at the top. You can enter the name, zip code, or other identifiers of your desired location. Upon selecting a location from those listed in your search, you return to the main screen. The button should now display your chosen location, and the images should soon reload to reflect the location change.

The location precision is limited to locality, so you can chose a country, state or province, city or town, but not street address. While you can select an International location, best results are found Domestically.



Bo

Boston
MA, United States

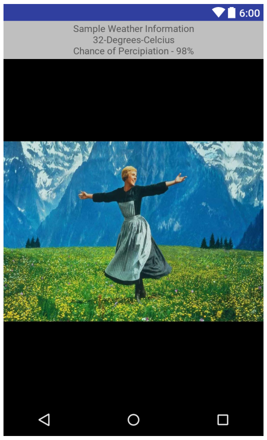
Bolton
MA, United States

Boylston
MA, United States

Boxborough

Expanding Images

Clicking on a weather image on the main screen, whether current or forecast, brings you to a new screen. This screen contains the image you clicked on, expanded to fit your screen, and specific weather information listed in a grey box above your image. Clicking on the expanded image or clicking on the return button on your device or emulation will return you to the main screen.



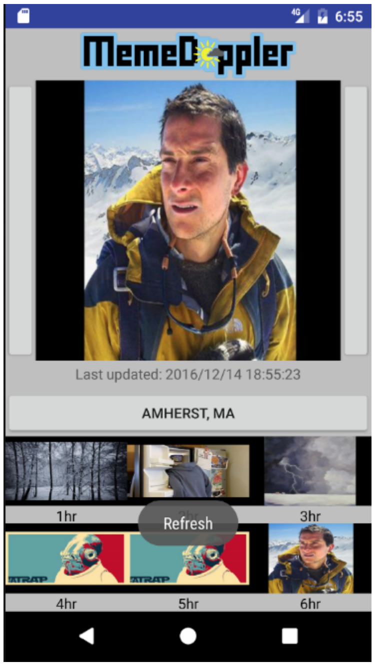
Sample Weather Information
32-Degrees-Celcius
Chance of Percipiation - 98%

Refreshing

There are many things the user can do to trigger a refresh of weather data and images. Swiping up or down, changing the location, and restarting the app all cause a refresh. Anything that causes the app to reload the main screen will also cause a refresh.

Refreshes change the images on the main screen,the images in the user feedback pane, and the images in the expanded screen. Refreshing also changes the specific weather information in the expanded screen.

However, if weather information has not significantly change due to time or location, refreshes may not change the displayed images. The time of the most recent refresh will be listed in the text above the location button.



Last updated: 2016/12/14 18:55:23

AMHERST, MA

Current State

Below are listed the differences between our current state and the Final State as listed above.

Main Screen

The basic functions and look of the main screen will remain virtually identical between the current state and final state. As it stands, our group is pretty happy with how the main screen functions.

User Feedback

Unfortunately, user feedback is one of the areas where we have had to cut back the most. As development of a user feedback system largely relies on functioning connections between the database and client. Until recently, we had no way to get data from the database. We currently cannot send data to the server, which is essential to providing user feedback. We have as a group decided to prioritize other incomplete features over user feedback for the final version of our app.

In other words, the motions users go through to provide feedback are implemented and functions as desired, but nothing can be done with the feedback at the moment.

Location

Location is fully functional, with a wide search range and ability to preserve location through refreshes and screen changes (way to go chad!).

Expanding Images

As of the writing of this document, we are not able to expanding images as desired. Our ability to affect other screens (activities) is limited, but I am very close to being able to send objects between activities. As we cannot send data from the main screen to the expanded screen, the images and weather data in the expanded image screen are hardcoded.

Refreshing

As of the writing of this document, we are not able to query for weather data or receive non-image data from the database. As such, the images that are being refreshed are hardcoded as a string array. Once we can server and database communication is implemented, it will be trivial to get refreshing working.

In other words, once we have data, our refreshing works as desired. We have the tools to find and replace images in the app with images from the server, without exceeding memory limits.