CS 4720 Final Project Report

Developer: Elaina Hill (emh2hb)

Device: No Device Used
Platform: Android

Project Title: Magic 8 Weather

Project Pitch

Magic 8 Weather is an application that gives users a detailed look at the weather for the day in a fun format. It's meant to mimic the style of a magic 8 ball as the user interacts with it. Loading the application will display the home screen, where a magic 8 ball and two buttons will appear. Clicking in the top right corner will display a drop-down menu where the user may select a city to view weather for or select a username for the app. Selecting a city will display the temperature and the name of the city as well as an icon in the middle of the magic 8 ball depicting what the weather looks like today. Depending on what the weather is like, the magic 8 ball (application) will tell you how today looks. For example, if the forecast is sunny, it will say, "The near future looks pretty sunny" and so on for various weather conditions. The application is meant to give users a more detailed look at their weather, which they can do by viewing 3 Hour or Next Day Forecasts, which give predictions on humidity, pressure, and a range of temperatures.

Platform Justification

The platform I chose for this project is Android, mainly because it is the platform that I feel most comfortable with in terms of usability and access to development material. Android studio, specifically, is a tool that I find easy to use since it has so many resources located in one place. The documentation for Android Studio at developer.android.com is not difficult to understand, which is important for beginner developers. I also like the fact that if I need to figure out how to do something, there is typically an example application that I can load up right away from Android Studio. Lastly, while I know other programming languages, I like the fact that I am able to use Java when developing for the Android platform since Java is the language that I have the most experience with overall.

Key Features

Current Weather

Users can quickly view current weather conditions by loading the application. Basic information is shown on the home screen after selecting a city in the settings menu. An icon will appear that will show basic weather conditions and the magic 8 ball will appear to tell you something about today's weather. Current temperature and the city selected will also appear on the screen as well as the time when that data was last updated.

3 Hour Forecasts

In the bottom left of the home screen, the application displays a "3 Hour Forecast" button. Clicking on this button will show the user how the weather looks for the next 3 hours in the city that they previously selected. It displays detailed information, such as humidity, pressure, general weather conditions, and minimum/maximum temperatures. These details you wouldn't usually get from a basic weather application that might show only the current temperature and overall conditions.

Next Day Forecasts
In the bottom right of the home screen, the application displays a "Tomorrow's Forecast" button. Clicking here will show the user what they should expect the weather to be like for the next day in the city that they selected most recently. Weather information for this screen includes general weather condition, daytime temperatures, evening temperatures, and nighttime temperatures.

Testing Methodologies

The application was tested in pieces during development. Before working on the second and third activities, I made sure that the first activity was in working order. Testing included making sure that I was receiving the correct data from my endpoints by entering many different cities and checking the JSON data by hand before parsing. I followed the same process for the second and third activities. In some cases, I had trouble figuring out if certain methods were being called and in those cases I also used 'toasts' to help me figure out what was going on in the application as I ran it many times using the Android Studio emulator.

When implementing local data storage, I testing closing the app and reopening again to make sure that data would persistent for both the city and username selected by the user. I made sure that each activity also kept the appropriate data after being restarted and that they displayed the correct information after the city had been changed.

Usage/Special Information

When running the application, you should select the city you wish to view weather for first. When selecting a city, enter it with no spaces (including after a comma, if specifying state or country). For example, if you want to view weather for Charlottesville, VA, the input should look like "Charlottesville" or "Charlottesville,va". Entering a zip code instead of a city name will show current weather, but no weather data will be found for the 3 Hour or Next Day Forecasts.

Lessons Learned

Local Storage

I learned how to keep data saved in an application by using Android's Shared Preferences. Shared Preferences will store information in the form of kay-value pairs and it can be accessed by all activities in an application. I implemented a separate class for storing and updating data for my application, which is how I was able to save a user's selected city and their screen name for the next time that they use the app. I had trouble at first with preferences being stored separately for each activity, but solved this problem by changing how I called the preferences from different activities. Being able to store data makes for better usability and a more personalized application experience that I appreciate.

Web Service Implementation

The project required me to learn how to gather data from existing web services for use with my application. In order to do this, I had to make sure permissions were specified within the manifest file and designate different classes to request weather data from OpenWeatherMap for each of my activities. I obtained an API key from openweathermap.org and, sent requests using HTTP, and converted API responses to JSON. I picked out each part of the JSON Object that I wanted to appear for the user and set it to a Text View. This was my first attempt at integrating web services for a mobile app.

Menu/Layouts/Other

While web services and local storage are the two major new things I learned about, I also learned how to do several smaller things. I implemented a drop-down menu for changing settings and learned how to change background and toolbar colors. I finally figured out how to use a custom picture set to an ImageView and how to change the background color of that image, even though I did not end up doing that.

Wireframe

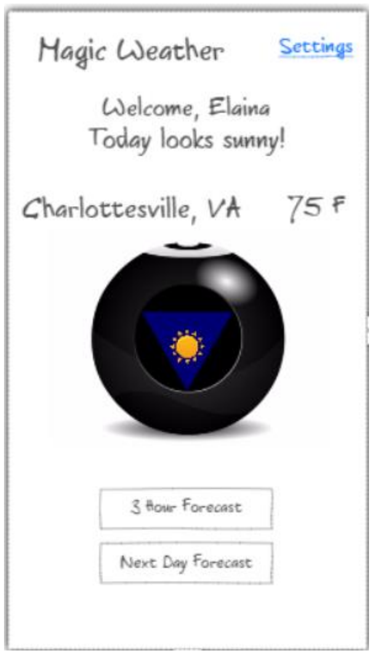

Image 1 depicts how the home screen should look.

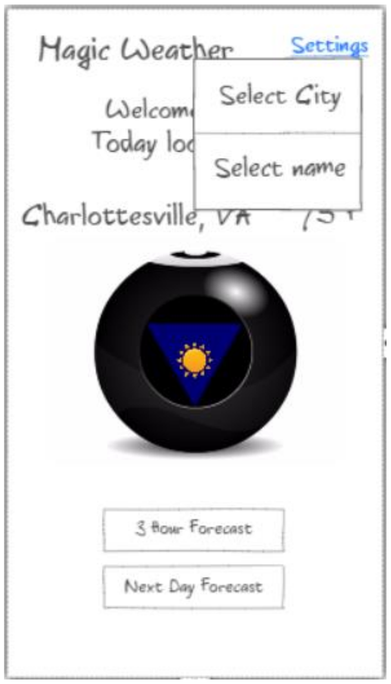

Image 2 shows how to access settings.

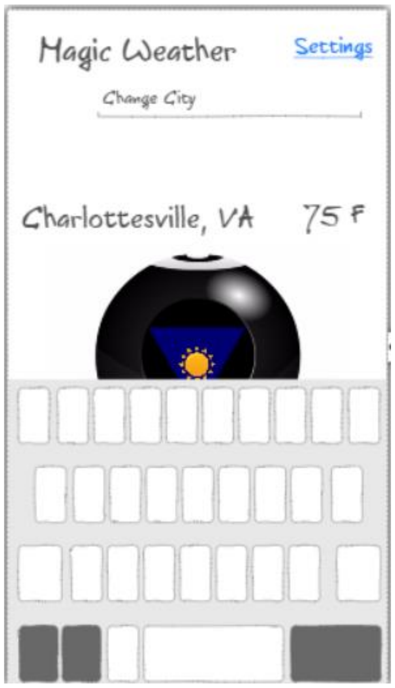

Image 3 depicts the process of changing cities.

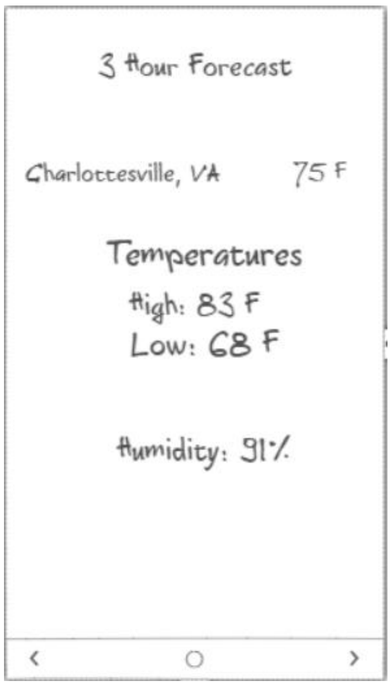

Image 4 represents a popup of the second activity.

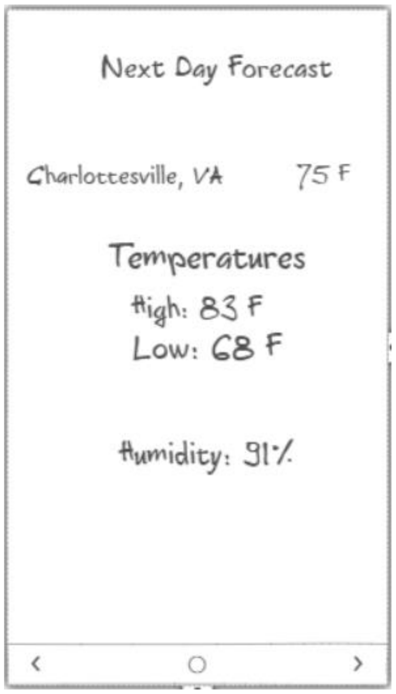

Image 5 represents a popup of the last activity.