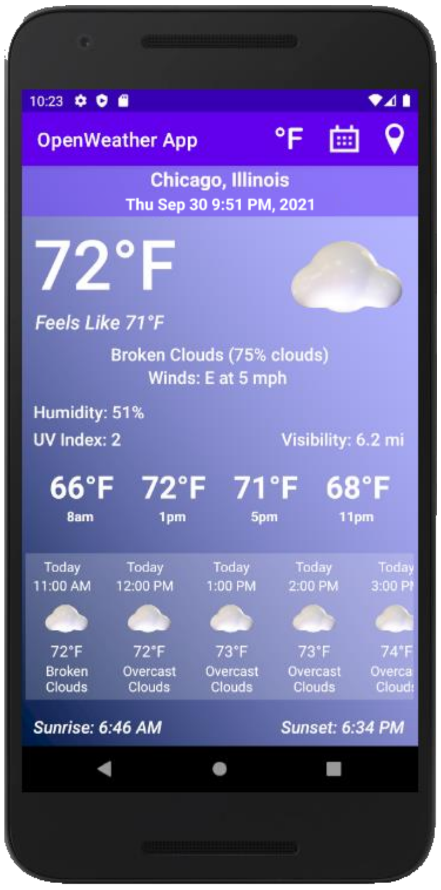
CS 442: Mobile Applications Development
Assignment 3 – Open Weather *(300 pts)*

Uses: Multiple Activities, Internet API, RecyclerViews, Option-Menus, Android Volley, JSON Data, Time conversion, Dialogs, Images

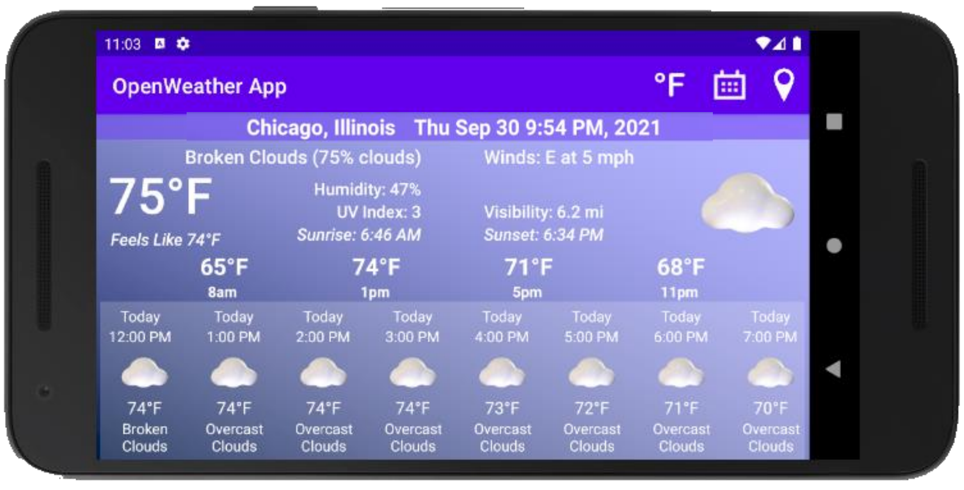
App Highlights:

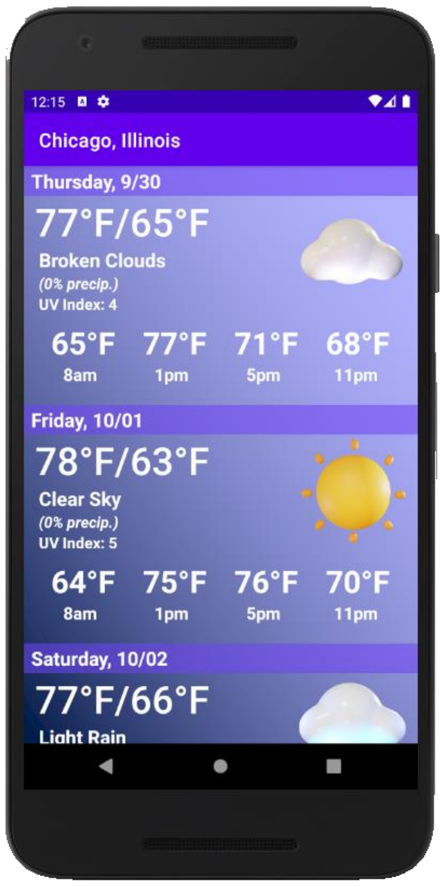
- This app displays a variety of weather data for a specified location – current weather, hourly forecast (48 hours), and daily forecast (7 days).
- The app is made up of 2 activities – the home weather screen, and the daily forecast screen.
- The home weather screen *will* need an alternate layout for landscape orientation (see images). The daily forecast screen *will not* need an alternate landscape layout – the same layout should work in any orientation.
- The data that makes up the current weather, hourly forecast, and daily forecast are detailed in the images in this document.
- The units (C/F) can be toggled between imperial and metric by tapping an options-menu icon.
- The daily forecast is displayed by tapping the calendar icon in the options-menu.
- The location for the displayed weather can be changed by tapping the location icon in the options-menu.



Home screen Portrait & Landscape layouts

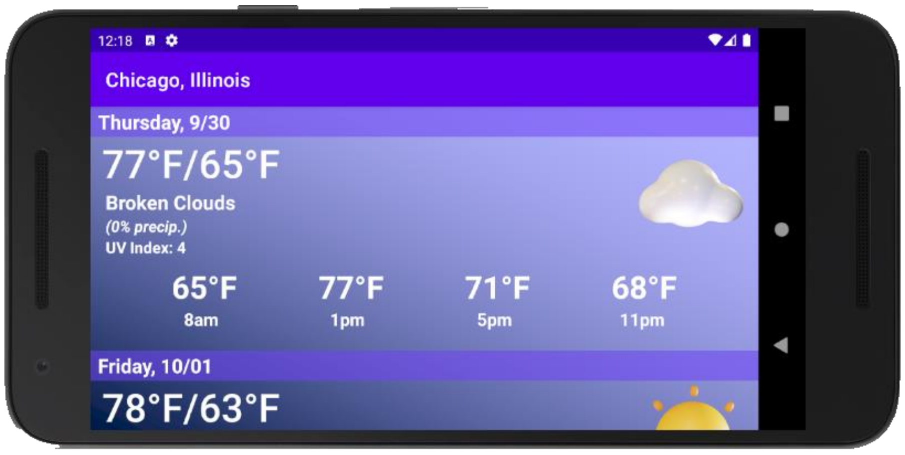
Content details are shown later


11:03
OpenWeather App
°F
Chicago, Illinois Thu Sep 30 9:54 PM, 2021
Broken Clouds (75% clouds)
Winds: E at 5 mph
75°F
Humidity: 47%
UV Index: 3
Visibility: 6.2 mi
Feels Like 74°F
Sunrise: 6:46 AM
Sunset: 6:34 PM
65°F 8am
74°F 1pm
71°F 5pm
68°F 11pm
Today 12:00 PM 74°F Broken Clouds
Today 1:00 PM 74°F Overcast Clouds
Today 2:00 PM 74°F Overcast Clouds
Today 3:00 PM 74°F Overcast Clouds
Today 4:00 PM 73°F Overcast Clouds
Today 5:00 PM 72°F Overcast Clouds
Today 6:00 PM 71°F Overcast Clouds
Today 7:00 PM 70°F Overcast Clouds

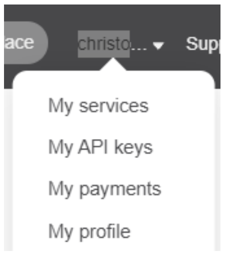


7-Day weather Portrait & Landscape layouts

Content details are shown later



A) Internet Data:



Weather data for this app will come from OpenWeather (https://openweathermap.org).

You will need to set up a free account and get an api key to use this API. You can create your account at https://home.openweathermap.org/users/sign_up.

Once you have your account and have logged in, click on your username in the upper-right and select “My API Keys”. From there you can access and copy your API Key. Your free account has a limit of 60 calls/minute, and a total of 1,000,000 calls/month.

We will use the *OneCall* API endpoint (documentation at https://openweathermap.org/api/one-call-api).

That call will look like:

```
https://api.openweathermap.org/data/2.5/onecall?lat=#########&lon=#########
         &appid=your_api_key&units=units_selection&lang=en&exclude=minutely
```

- lat: decimal latitude value *(use 41.8675766 as the default latitude value)*
- lon: decimal latitude value *(use -87.616232 as the default latitude value)*
- appid: your openweathermap.org api key
- units: the selected units to be used in the response (imperial or metric)

The JSON results of this endpoint call look like the below. Data we will make use of is highlighted in yellow. The unhighlighted data can be ognored.

```
{
        "lat": 41.8676,
        "lon": -87.6162,
        "timezone": "America/Chicago",
        "timezone_offset": -18000,
        "current": {
                "dt": 1633012174,
                "sunrise": 1633002388,
                "sunset": 1633044845,
                "temp": 67.46,
                "feels_like": 66.54,
                "pressure": 1022,
                "humidity": 56,
                "dew_point": 51.24,
                "uvi": 0.8,
                "clouds": 75,
                "visibility": 10000,
                "wind_speed": 4.61,
                "wind_deg": 120,
                "wind_gust": 8.25,  ← optional, may not appear
                "weather": [{  ← If more than one object is present, only use the first one
                        "id": 803,
                        "main": "Clouds",
                        "description": "broken clouds",
                        "icon": "04d"
                }],
                "rain": {       ← "rain" section is optional, may not appear
                        "1h": 0.21
                },
                "snow": {       ← "snow" section is optional, may not appear
                        "1h": 0.21
                }

        },
        "hourly": [{  ← There will be many "hourly" objects here – only showing one here
                        "dt": 1633010400,
                        "temp": 67.46,
                        "feels_like": 66.54,
                        "pressure": 1022,
                        "humidity": 56,
                        "dew_point": 51.24,
                        "uvi": 0.8,
                        "clouds": 75,
                        "visibility": 10000,
                        "wind_speed": 6.6,
                        "wind_deg": 141,
                        "wind_gust": 8.55,
                        "weather": [{  ← If more than one object is present, only use the first one
                                "id": 803,
                                "main": "Clouds",
                                "description": "broken clouds",
                    "icon": "04d"
                }],
                "pop": 0
            },
            {
            …
            }
        ],
        "daily": [{  🡐 There will be many “daily” objects here – only showing one here
                "dt": 1633021200,
                "sunrise": 1633002388,
                "sunset": 1633044845,
                "moonrise": 1632978060,
                "moonset": 1633035060,
                "moon_phase": 0.8,
                "temp": {
                    "day": 70.57,
                    "min": 65.89,
                    "max": 73.96,
                    "night": 68.11,
                    "eve": 71.35,
                    "morn": 66.04
                },
                "feels_like": {
                    "day": 69.73,
                    "night": 67.44,
                    "eve": 70.72,
                    "morn": 65.35
                },
                "pressure": 1022,
                "humidity": 51,
                "dew_point": 51.58,
                "wind_speed": 8.05,
                "wind_deg": 78,
                "wind_gust": 12.77,
                "weather": [{ 🡐 If more than one object is present, only use the first one
                    "id": 804,
                    "main": "Clouds",
                    "description": "overcast clouds",
                    "icon": "04d"
                }],
                "clouds": 90,
                "pop": 0,
                "uvi": 4.17
            },
            {
            …
            }
        ]
    }
```

B) Application/Behavior Diagrams:

1) App Main screen

Options Menu (3 items):
- *Toggle units*
- *Show daily forecast*
- *Change location*

Location name from Geocoder (process described later in this document).

Current Date/Time (current.dt)

Temperature (current.temp)

Feels Like (current.feels_like)

Humidity (current.humidity)

UV Index (current.uvi)

Morning Temp (daily[0].temp.morn)

Daytime Temp (daily[0].temp.day)

48 Hourly Weather RecyvlerView (see below)

Sunrise (current.sunrise)

Weather Icon (current.weather.icon)

Weather description (current.weather.description)

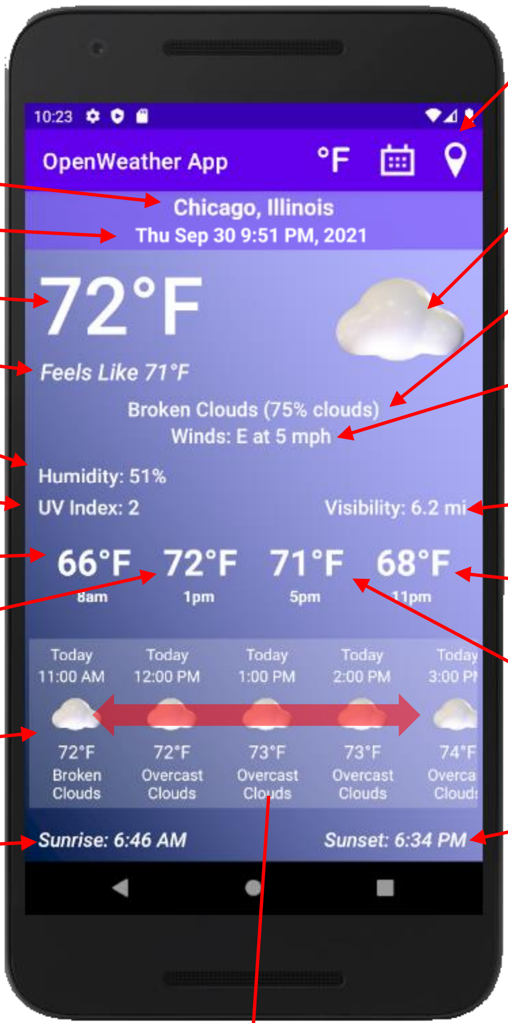
Winds (current.wind_speed, current_wind_deg, current.wind_gust)

Visibility (current.visibility

Night Temp (daily[0].temp.night)

Evening Temp (daily[0].temp.eve)

Sunset (current.sunset)


10:23
OpenWeather App °F
Chicago, Illinois
Thu Sep 30 9:51 PM, 2021
72°F
Feels Like 71°F
Broken Clouds (75% clouds)
Winds: E at 5 mph
Humidity: 51%
UV Index: 2
Visibility: 6.2 mi
66°F 8am | 72°F 1pm | 71°F 5pm | 68°F 11pm

Today 11:00 AM	Today 12:00 PM	Today 1:00 PM	Today 2:00 PM	Today 3:00 PM
72°F Broken Clouds	72°F Overcast Clouds	73°F Overcast Clouds	73°F Overcast Clouds	74°F Overcast Clouds

Sunrise: 6:46 AM
Sunset: 6:34 PM

Daily Weather RecyclerView daily entries.

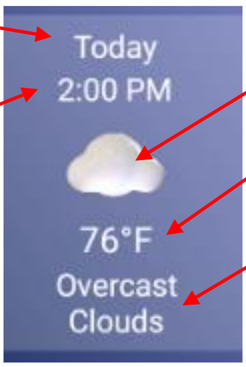
Day ("Today", "Day name", "Day name", etc.). For example: "Today", "Friday", "Saturday". (hourly[n].dt)

Time HH:MM am/pm (hourly[n].dt)

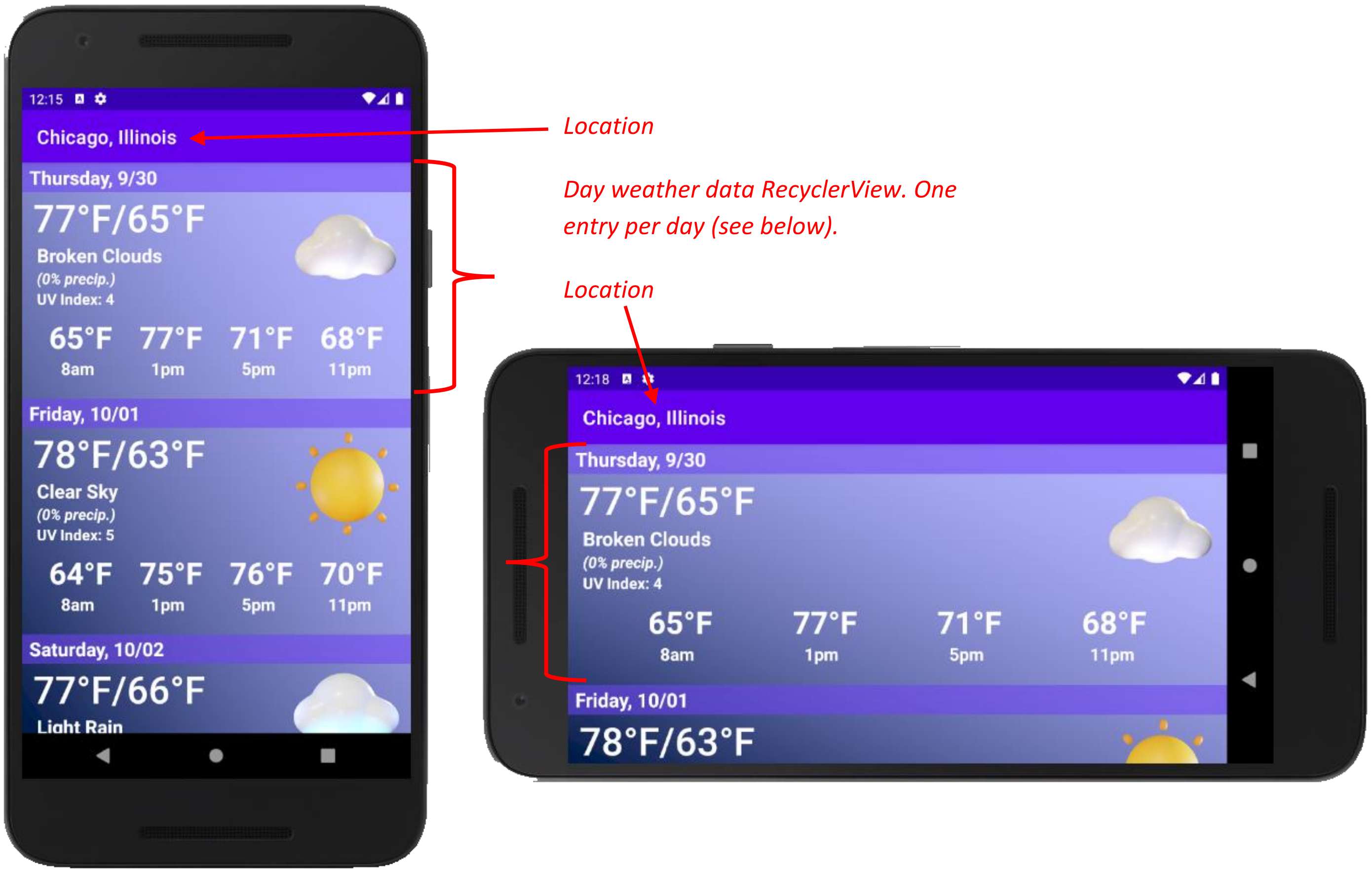
Weather icon (hourly[n].weather.icon)

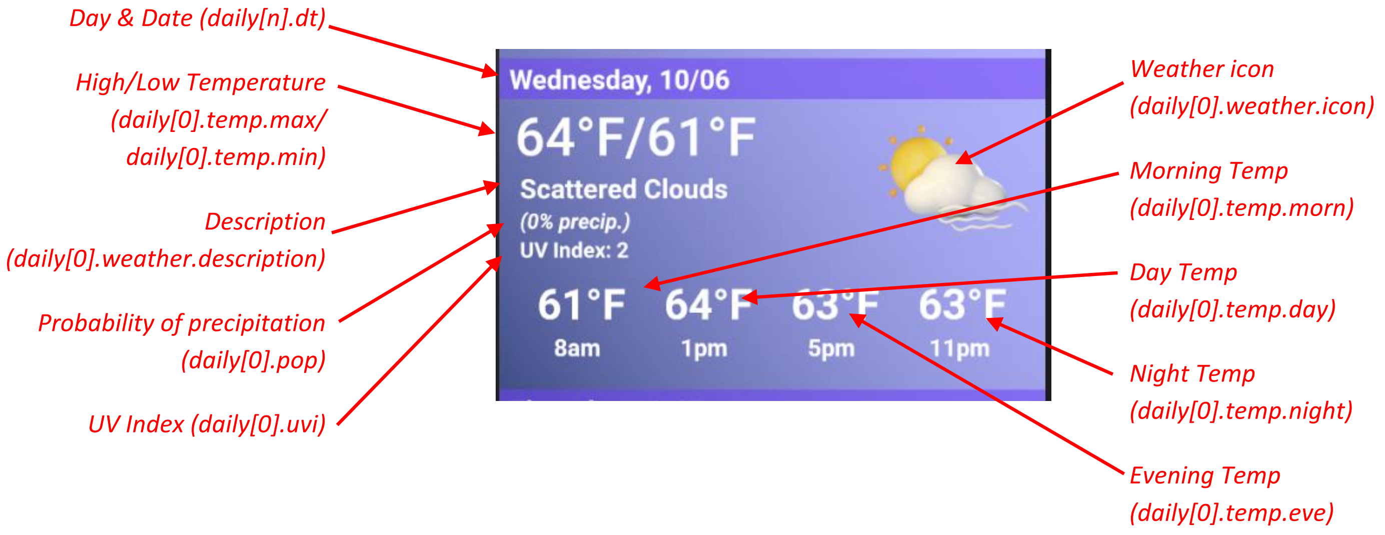
Temperature (hourly[n].temp)

Description {hourly[0].weather.description)

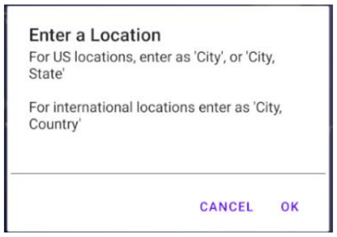

Today
2:00 PM
76°F
Overcast Clouds

2) 7-Day weather Portrait & Landscape




Day & Date (daily[n].dt)
High/Low Temperature (daily[0].temp.max/ daily[0].temp.min)
Description (daily[0].weather.description)
Probability of precipitation (daily[0].pop)
UV Index (daily[0].uvi)
Wednesday, 10/06
64°F/61°F
Scattered Clouds
(0% precip.)
UV Index: 2
61°F 64°F 63°F 63°F
8am 1pm 5pm 11pm
Weather icon (daily[0].weather.icon)
Morning Temp (daily[0].temp.morn)
Day Temp (daily[0].temp.day)
Night Temp (daily[0].temp.night)
Evening Temp (daily[0].temp.eve)

3) Options Menu
   - Change Units
     Selecting this menu item should toggle the selected measurement unit from imperial to metric or from metric to imperial. When changed, the OneCall API endpoint should be called again, specifying the newly selected unit. The change unit menu item's icon should be changed to match the selected unit (°F for imperial, °C for metric – icon images provided)
   - Daily Forecast
     Selecting this menu item should open the daily forecast activity (passing the daily forecast data content).
   - Change Location
     Selecting this menu item should display an AlertDialog that allows the user to enter a new location, as shown below. Upon tapping OK, the specified location should be used with a GeoCoder to get the latitude and longitude (see section 6 for details). The latitude and longitude should then be used to call the OneCall API endpoint to get the new weather data.


Enter a Location
For US locations, enter as 'City', or 'City, State'

For international locations enter as 'City, Country'

4) Weather icons
   Weather icons found in the JSON weather data are provided as a string code (04d, 10n, etc). These codes correspond to image file names (image files are provided to you). These icon names are used to access these provided images within your project (in the drawable resource folder). The following code shows how to do this:

   // prepend an underscore to the icon code (i.e., "04d" becomes "_04d", "10n" becomes "_10n", etc))

```
iconCode = "_" + iconCode;
```

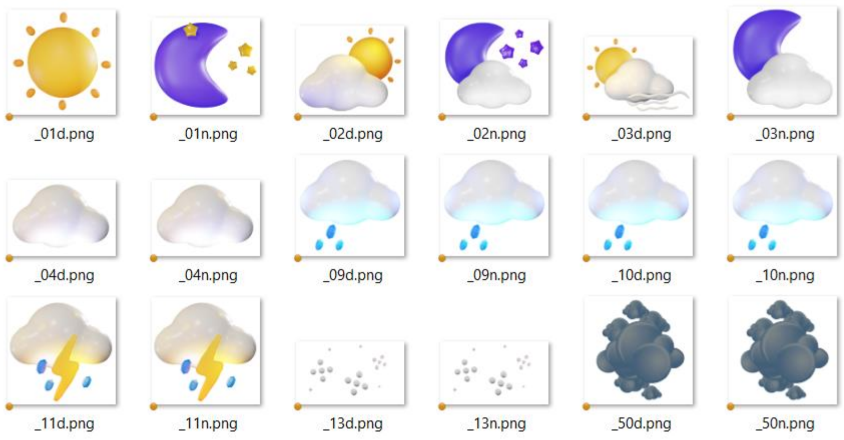
   // Then get the image resource id

```
int iconResId = someActivity.getResources().getIdentifier(icon code,
                                     "drawable", someActivity.getPackageName());
```

   // This resource id can then be used to set the image of an ImageView:

```
someImageView.setImageResource(iconResId);
```

   Provided Weather Icons:



5) Time Conversions

Date/Time (“dt”) values are provided as the current time, Unix, UTC. These need to be converted to Date/Time values in local time, then formatted as needed. Note, the “timezone_offset” value (from the JSON data) is needed to do this. This can be done as follows:

```
LocalDateTime ldt =
   LocalDateTime.ofEpochSecond(dt + timezone_offset(), 0, ZoneOffset.UTC);

DateTimeFormatter dtf =
   DateTimeFormatter.ofPattern("EEE MMM dd h:mm a, yyyy", Locale.getDefault());

String formattedTimeString = ldt.format(dtf); // Thu Sep 30 10:06 PM, 2021
```

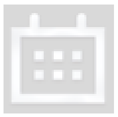
Please see https://docs.oracle.com/javase/8/docs/api/java/time/format/DateTimeFormatter.html for more details on the DateTimeFormatter, as other Date and Time strings will be needed in this application.

6) Getting a latitude/longitude using a GeoCoder

The latitude/longitude can be derived from the location name. The below function can also be used to do this.

```
private double[] getLatLon(String userProvidedLocation) {

    Geocoder geocoder = new Geocoder(this); // Here, “this” is an Activity
    try {
        List<Address> address =
                geocoder.getFromLocationName(userProvidedLocation, 1);

        if (address == null || address.isEmpty()) {
            // Nothing returned!
            return null;
        }
        lat = address.get(0).getLatitude();
        lon = address.get(0).getLongitude();

        return new double[] {lat, lon};

    } catch (IOException e) {
        // Failure to get an Address object
        return null;
    }
}
```

7) Additional provided images


daily.png

units_c.png

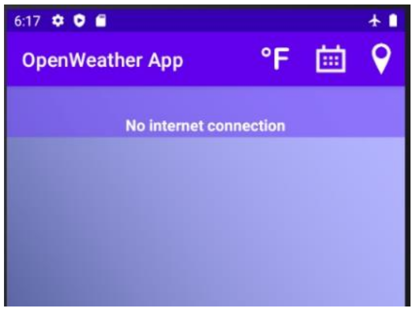
background.png

location.png

units_f.png

8) Handling no-network situations

When the device has no network connection, the app cannot access the OpenWeather API site (and the GeoCoders will not be able to function). In those situations (when the app attempts to access the internet without a connection), the home screen's data/time text view should show "No internet connection". When there is no network connection, the options-menu selections should *not* function. (Displaying a Toast message indicating that the function cannot be used when there is no network connection is fine in this situation).



No internet connection

```
private boolean hasNetworkConnection() {
    ConnectivityManager connectivityManager = getSystemService(ConnectivityManager.class);
    NetworkInfo networkInfo = connectivityManager.getActiveNetworkInfo();
    return (networkInfo != null && networkInfo.isConnectedOrConnecting());
}
```

Note – you must declare the ACCESS_NETWORK_STATE permission in the project manifest!

9) Converting Wind Direction

The wind direction is provided by the API in degrees (0-359 degrees). The app should display this in a more conventional fashion, using compass points (N, SE, W, etc). This can be derived by specifying what degrees correspond to these compass points. For purposes of this application, you can use the following:

```
private String getDirection(double degrees) {
    if (degrees >= 337.5 || degrees < 22.5)
        return "N";
    if (degrees >= 22.5 && degrees < 67.5)
        return "NE";
    if (degrees >= 67.5 && degrees < 112.5)
        return "E";
    if (degrees >= 112.5 && degrees < 157.5)
        return "SE";
    if (degrees >= 157.5 && degrees < 202.5)
        return "S";
    if (degrees >= 202.5 && degrees < 247.5)
        return "SW";
    if (degrees >= 247.5 && degrees < 292.5)
        return "W";
    if (degrees >= 292.5 && degrees < 337.5)
        return "NW";
    return "X"; // We'll use 'X' as the default if we get a bad value
}
```

10) Extra Credit Options

There are 2 extra credit options in this project – Incorporating "Swipe Refresh", and "Saving User Settings".

- (+15 pts) Swipe Refresh – Add swipe-refresh capability (to the home screen only) to reload the weather data for the current location. If there is no network connection, do nothing – leave the existing weather data in place and indicate the no-network situation in a Toast message.
- (+15 pts) Saving User Settings – Save the currently selected location latitude and longitude and the current units selection so that when the app is restarted, the weather displayed is for the saved location, and is displayed in the saved measurement units. Note, the "units" options-menu and all displayed temperature values should correctly display the currently selected measurement units.
- (+30) Open Calendar - Tapping a single entry in the hourly forecast RecyclerView (on the main screen) should open the calendar app to that date/time (theoretically, this would allow the user to view any appointments are that date/time).

Assignment Assistance

The TAs for our course are available to assist you with your assignment if needed. Questions on assignment requirements and course concepts can be sent to the instructor.

Submissions & Grading

1) Submissions must consist of your zipped project folder *(please execute Build =>Clean Project before generating the zip file*).
2) Submissions should reflect the concepts and practices we cover in class, and the requirements specified in this document.
3) Late submissions will be penalized by 10% per week late. (i.e., from one second late to 1 week late: 10% penalty, from one week late to 2 weeks late: 20% penalty, etc.). NO SUBMISSIONS CAN BE MADE BEYOND 2 WEEKS LATE.
4) Grading will be based upon the presence and proper functionality of *all features and behaviors* described in this document. NOTE: All descriptions and images constitute the requirements for this assignment.

NOTE

This assignment is worth 300 points. This means (for example) that if you get 89% on this assignment, your recorded score will be:

(89% * 300 points = 267 points)

Note that this also means that the 10% late submission penalty will be 10% * 300 points = 30 points.

If you do not understand anything in this handout, please ask.

Otherwise, the assumption is that you understand the content.

Unsure? Ask!