Chapter Twelve Activity: Weather Application

Updated July, 2017

In this activity, we will use a network connection to get current weather data from an online weather source!

Project Details

This program will allow the user to select their location and then will show them the current weather, tomorrow's weather and the next day's weather. We will retrieve the weather information from a free weather service on the Internet.

World Weather Online

For this activity, we will be using a data feed from World Weather Online. This company provides free weather information in various formats to programmers.

Please CONTACT US at the link below to request a copy of the **free registration key** needed to use the World Weather Online service.

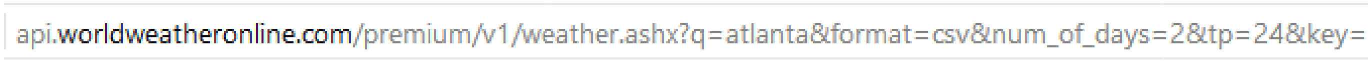
http://www.compuscholar.com/homeschool/contact-us/

Weather Data Stream

In order to get our weather data, we will make a network connection to a specific URL on the World Weather Online website. The exact URL will tell the weather service our location and what type of data we are requesting. In our case, we will pass the user's location and a request for two-day weather, which will give us the current weather and weather for the next two days.

You can retrieve data from an Internet server in many different ways. We will request our data as a "comma-delimited string". This means that individual pieces of data will be separated by commas in a larger string. This is a very common and simple way to retrieve data from a server.

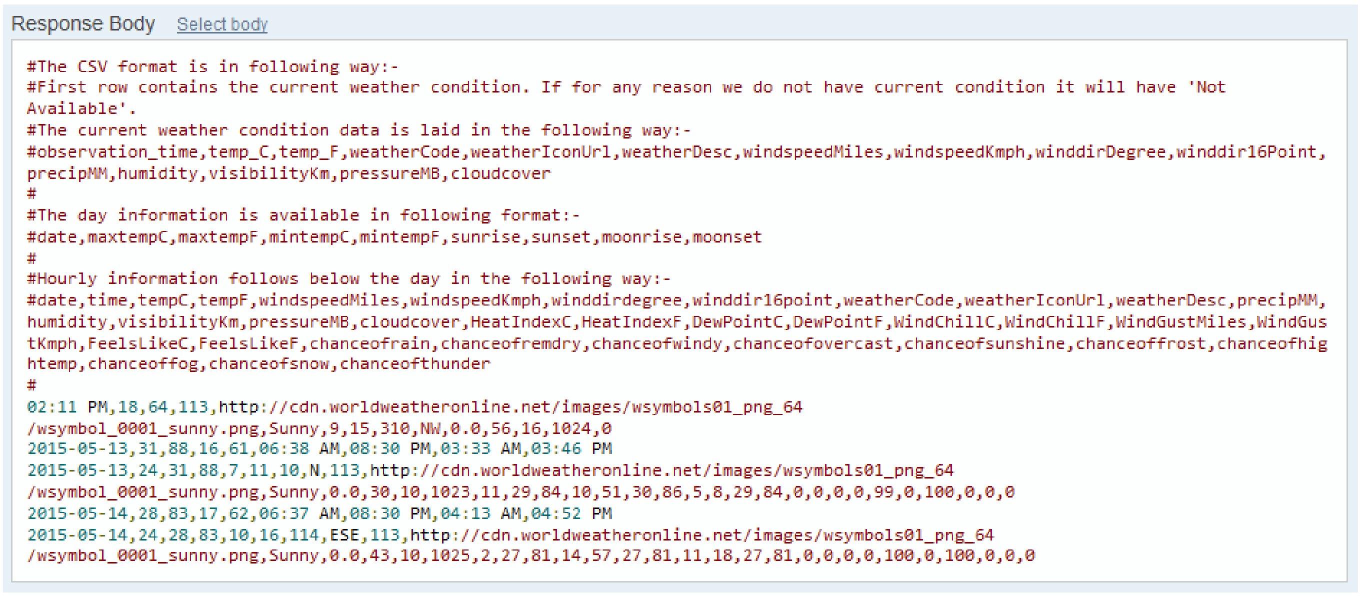
To request comma-delimited data, your program will use an URL similar to the one shown below. The registration key will appear at the end (not shown below).



```
api.worldweatheronline.com/premium/v1/weather.ashx?q=atlanta&format=csv&num_of_days=2&tp=24&key=
```

You can see the URL starts with the text "**http://api.worldweatheronline.com/premium/v1/weather.ashx?q=**" and then we see our city: "Atlanta". After that is some information about the data we want returned: "&format=csv". This tells the server to send us the data in "comma-separated values" or "csv". After that is "&num_of_days=2", which tells the server to send us two-day weather information. The "tp=24" parameter tells the service to just give us one line of details per day, and avoid hourly information. Your API key will appear at the end.

The example response data shown below demonstrates the comma-separated lines of text that can be returned from your URL request. The return data will actually be one long string, separated by new-line characters.



```
#The CSV format is in following way:-
#First row contains the current weather condition. If for any reason we do not have current condition it will have 'Not
Available'.
#The current weather condition data is laid in the following way:-
#observation_time,temp_C,temp_F,weatherCode,weatherIconUrl,weatherDesc,windspeedMiles,windspeedKmph,winddirDegree,winddir16Point,
precipMM,humidity,visibilityKm,pressureMB,cloudcover
#
#The day information is available in following format:-
#date,maxtempC,maxtempF,mintempC,mintempF,sunrise,sunset,moonrise,moonset
#
#Hourly information follows below the day in the following way:-
#date,time,tempC,tempF,windspeedMiles,windspeedKmph,winddirdegree,winddir16point,weatherCode,weatherIconUrl,weatherDesc,precipMM,
humidity,visibilityKm,pressureMB,cloudcover,HeatIndexC,HeatIndexF,DewPointC,DewPointF,WindChillC,WindChillF,WindGustMiles,WindGus
tKmph,FeelsLikeC,FeelsLikeF,chanceofrain,chanceofremdry,chanceofwindy,chanceofovercast,chanceofsunshine,chanceoffrost,chanceofhig
htemp,chanceoffog,chanceofsnow,chanceofthunder
#
02:11 PM,18,64,113,http://cdn.worldweatheronline.net/images/wsymbols01_png_64
/wsymbol_0001_sunny.png,Sunny,9,15,310,NW,0.0,56,16,1024,0
2015-05-13,31,88,16,61,06:38 AM,08:30 PM,03:33 AM,03:46 PM
2015-05-13,24,31,88,7,11,10,N,113,http://cdn.worldweatheronline.net/images/wsymbols01_png_64
/wsymbol_0001_sunny.png,Sunny,0.0,30,10,1023,11,29,84,10,51,30,86,5,8,29,84,0,0,0,0,99,0,100,0,0,0
2015-05-14,28,83,17,62,06:37 AM,08:30 PM,04:13 AM,04:52 PM
2015-05-14,24,28,83,10,16,114,ESE,113,http://cdn.worldweatheronline.net/images/wsymbols01_png_64
/wsymbol_0001_sunny.png,Sunny,0.0,43,10,1025,2,27,81,14,57,27,81,11,18,27,81,0,0,0,0,100,0,100,0,0,0
```

The first several lines all start with a "#" character. This means that these lines are comment lines and do not contain real data. Notice the comments will describe the order of the data that is provided.

The first data line will contain information about the current weather:

```
02:11 PM,18,64,113,http://cdn.worldweatheronline.net/images/wsymbols01_png_64/
wsymbol_0001_sunny.png,Sunny,9,15,310,NW,0.0,56,16,1024,0
```

The comma-delimited values have the following meanings (in this order):

- Observation Time – the time that the current weather was observed by the weather service
- Current Temperature – the current temperature in Celsius
- Current Temperature – the current temperature in Fahrenheit
- Weather Code – a 3 digit code that refers to a common weather table
- Weather Icon URL – the URL for the image which shows the current conditions in a visual format
- Weather Description – a short description of the current conditions
- Wind speed – the current wind speed value (in miles)
- Wind speed – the current wind speed value (in km)
- Wind Direction – the current wind direction in degrees
- Wind Direction 16pt – the current wind direction in "N", "NE", "S", "SSE", etc format
- Precipitation – the current precipitation in millimeters
- Humidity – the current humidity level
- Visibility – the current visibility in miles
- Pressure – the current barometric pressure level
- Cloud Cover – the current cloud cover status

The forecast information for the two requested days comes in a pair of lines for each day. The first line contains some summary information:

```
2015-05-13,31,88,16,61,06:38 AM,08:30 PM,03:33 AM,03:46 PM
```

…and the following line contains additional details:

```
2015-05-13,24,31,88,7,11,10,N,113,http://cdn.worldweatheronline.net/images/
wsymbols01_png_64/wsymbol_0001_sunny.png,Sunny,0.0,30,10,1023,11,29,84,10,51,30,8
6,5,8,29,84,0,0,0,0,99,0,100,0,0,0
```

You can review the # comments in the response body to see exactly what each field contains. We will not be using all of this information in our application. Instead we will pick a few key pieces of information to display to the user.

The Starter Project

We have created a starter project for you with two pre-defined activities, **Main** and **Configure**, the corresponding XML layout files, and the "AndroidMainfest.xml".

Please use the updated starter project found on our TeenCoder: Java/Android installation page:
http://www.compuscholar.com/homeschool/courses/textbooks/installation/java/

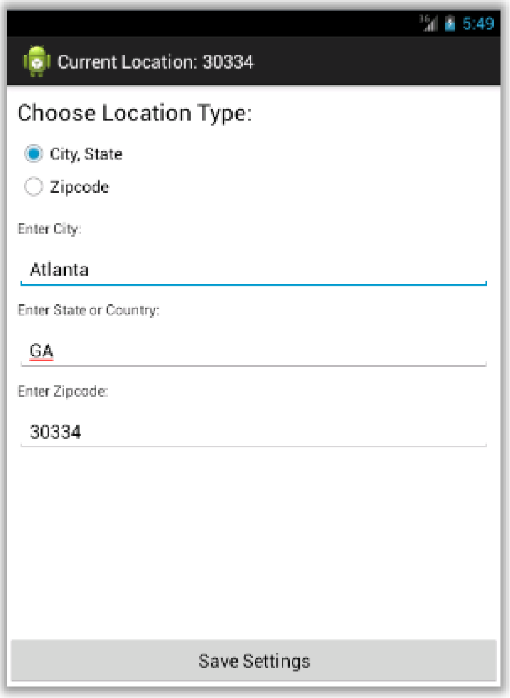
Here is the direct download link for the starter project in ZIP Format:
Updated Chapter 12 activity starter project - (ZIP)

Save this ZIP file to your "Activity Starters" directory and un-zip it to get the updated starter code. Then, copy this new folder to your "My Projects" directory and import the project into your Eclipse workspace in order to begin work on this project.

The WeatherInfo Class

The starter project also contains a utility class called **WeatherInfo**. This class will convert the comma-delimited weather data strings into an object that contains easy-to-read properties such as temperature and description. You won't have to deal with this class right away since the data processing and UI-handling code is already completed for the main screen. In later projects you'll make use of it yourself.

Activity Screens


5:49
Current Location: 30334
Choose Location Type:
● City, State
○ Zipcode
Enter City:
Atlanta
Enter State or Country:
GA
Enter Zipcode:
30334
Save Settings

This program has two screens: a configuration screen (**Configure**) and a weather display screen (**Main**).

The configuration screen will allow you to choose your location. You can specify this information by city and state, or by zip code.

Note: The zip code field will also work for global post codes and the city and state can be used to enter a city and country as well.

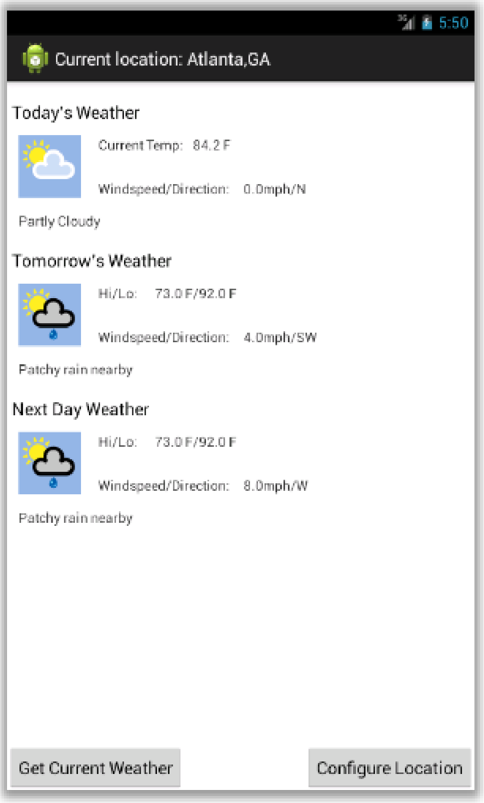
When the user clicks the "Save Settings" button, the location information is saved to the shared preferences location for this application.

The weather screen will display several pieces of useful weather information. The first information that we will display is the current location in the title bar for the both screens.

For the current day, we will display an image that demonstrates the current conditions. Under this image, we will display the text for the current conditions. To the right of the image, we will show the current temperature and wind conditions.

For tomorrow and the next day weather, we will again display images that represent the expected conditions and a text description of the conditions. To the right of this information, we will display the high and low temperatures and expected wind information.

At the bottom of the screen, we will have two buttons. The "Get Current Weather" button will refresh the current weather on the screen. The "Configure Location" button will show the configuration screen again and allow the user to change their current location.


Current location: Atlanta,GA

Today's Weather
Current Temp: 84.2 F
Windspeed/Direction: 0.0mph/N
Partly Cloudy

Tomorrow's Weather
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 4.0mph/SW
Patchy rain nearby

Next Day Weather
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 8.0mph/W
Patchy rain nearby

How to Complete this Project

To complete this project, you will need to complete these main tasks:

- Get the free API Key provided by your Homeschool Programming support request.
- In the **Main** class, enter your API key into the the **APIKey** variable:

```
    final String APIKey = "<your key here>";
```

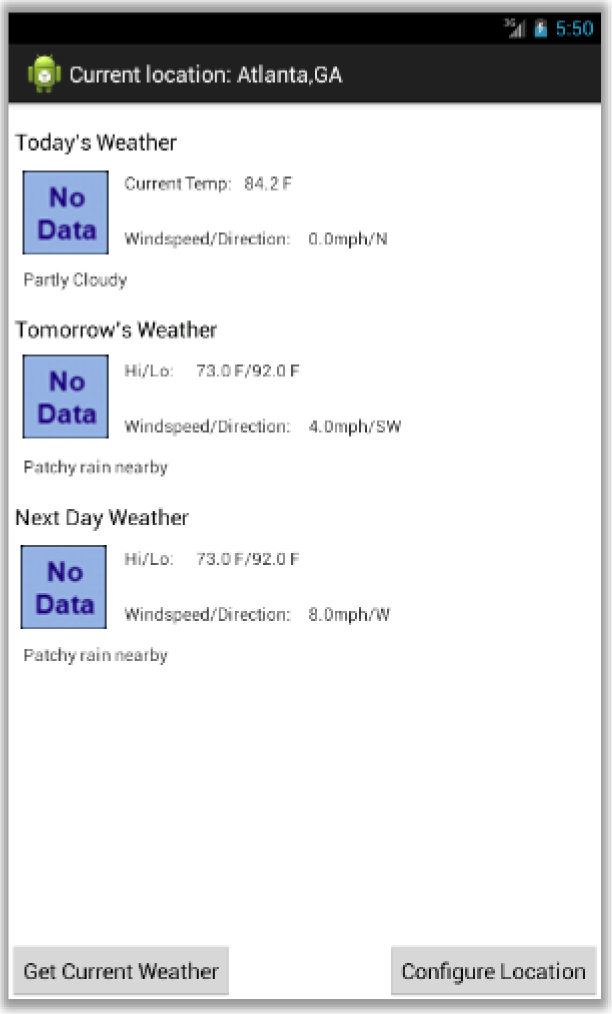
- Complete the **downloadWeather**() method in the **Main** class
- Complete the **downloadWeatherImage**() method in the **Main** class

Complete the downloadWeather() Method

In the **Main** class, find the **downloadWeather**() method. You will implement this method to retrieve an **ArrayList** of **Strings** containing comma-delimited weather information. This method is defined as a **public static** method because we will eventually be using it from other classes!

In this method you will need to:

- Create a new **ArrayList<String>** to hold your results.
- Check to see if the device has network connectivity with **Main.isNetworkAvailable**(), passing in your **`context`** parameter to the **downloadWeather**() method. Return immediately if the network is not available (your **ArrayList** object will not have any lines of text).
- To get the URL string to connect to the weather service based on your location, call the static **Main.getWeatherURL**() method and pass in the **`location`** parameter.
- Create a new **URL** object with the address string that is returned by **Main.getWeatherURL**()
- Make an HTTP connection for a "Get" operation.
- Connect to the server and get the response code. If the response is good, use an **InputStream, InputStreamReader,** and **BufferedReader** as demonstrated in the textbook to read lines of text from the server. For each line:
    - If the line starts with a "#" character, disregard the line. This is just a comment line.
    - If the line does not start with a "#" character, add the line of text to your **ArrayList<String>** results object.
- Remember to close the **BufferedReader** and **HttpURLConnection** objects when done
- Remember to wrap all risky network code with a **try/catch** block.
- Return your **ArrayList<String>** object at the end!


5:50
Current location: Atlanta,GA

Today's Weather
No Data
Current Temp: 84.2 F
Windspeed/Direction: 0.0mph/N
Partly Cloudy

Tomorrow's Weather
No Data
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 4.0mph/SW
Patchy rain nearby

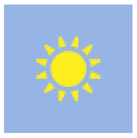
Next Day Weather
No Data
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 8.0mph/W
Patchy rain nearby

Get Current Weather
Configure Location

Checkpoint: When you build and run your program now, you should be able to configure your current location and then view the weather information. You have not yet completed the code to download the weather image, so you will see "No Data" where the images will be placed.

Complete the downloadWeatherImage() Method

The incoming data from the weather server will include a URL that can be used to retrieve the correct weather image from the server. The URL will look something like this:



http://cdn.worldweatheronline.net/images/wsymbols01_png_64/wsymbol_0001_sunny.png

You can use this URL to download this image and show it in your program! To do this, you will need to download binary data using a **BitmapFactory** to create a **Bitmap** as described in your student textbook. The bitmap can then be used to set the current image in your **ImageView** controls.

Most of the UI-handling code is already done for you in the starter project. You will need to complete the **downloadWeatherImage**() method which receives a URL string to an image and should return a **Bitmap**. This method is again **public static** because we'll want to re-use it in a later project from another class.

```
public static Bitmap downloadWeatherImage(Context context, String IconURL)
```

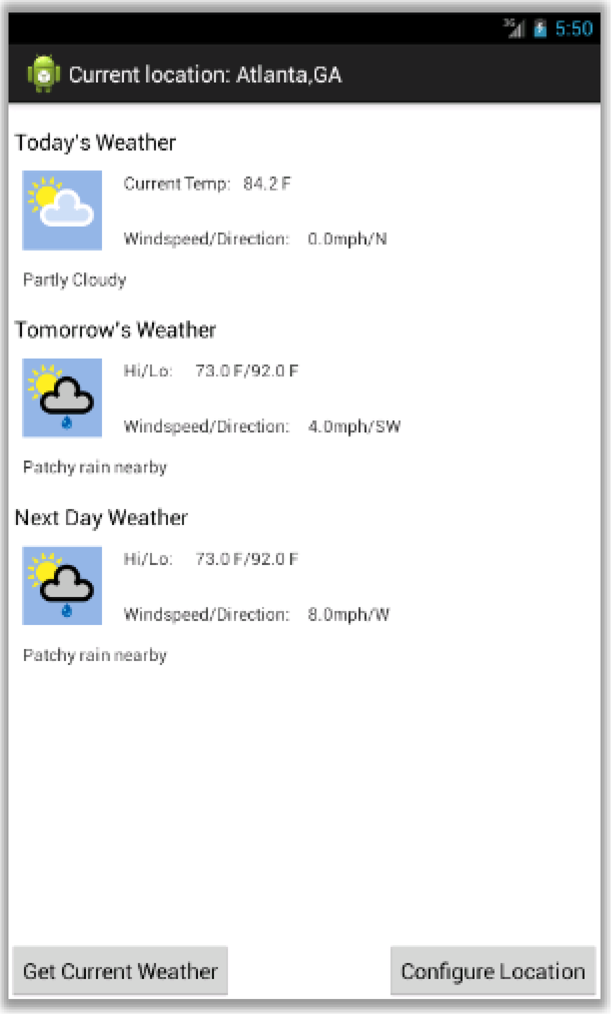
In the **downloadWeatherImage**() method, take the following steps:

- Check to see if the device has network connectivity with **Main.isNetworkAvailable**(). If not, return **null** immediately from the method.
- Create a new **URL** object with the **IconURL** string parameter to this method
- Make an HTTP connection for a "Get" operation.
- Connect to the server and get the response code. If the response is good, use an **InputStream** and **BitmapFactory** as demonstrated in the textbook to create a **Bitmap** object.
- Remember to close the **InputStream** and **HttpURLConnection** objects when done
- Remember to wrap all risky network code with a **try/catch** block.
- Return the **Bitmap** object to the calling program at the end, or **null** if unsuccessful for any reason.

This method will be called automatically by the existing UI-handling code in the starter project. Once you finish it successfully, you should be able to run your program and see weather images for each day!

Project Output

Your weather application is now complete! You should see full weather data and images for the current and next two days. You should be able to configure different locations and see different weather information.


5:50
Current location: Atlanta,GA

Today's Weather
Current Temp: 84.2 F
Windspeed/Direction: 0.0mph/N
Partly Cloudy

Tomorrow's Weather
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 4.0mph/SW
Patchy rain nearby

Next Day Weather
Hi/Lo: 73.0 F/92.0 F
Windspeed/Direction: 8.0mph/W
Patchy rain nearby