Weather Forecaster

Varun Guwal, SY I.T., Division – A, Roll no. – A009, SAP ID – 45207220001

Abstract/Synopsis:

I have developed a program to display the current weather, as well as the forecasted weather for the next five days, from a website called OpenWeatherMap and World Weather Information Service. The weather data is displayed within an application developed using Java Swing. It is updated every time I execute the program.

Keywords:

Weather, Forecasting, Weather Forecasting, Temperature, Celsius, Visibility, Clouds, Minimum Temperature, Maximum Temperature, Pressure, Humidity

Objective and Scope:

The main objective of this program is to display the current weather information as well as the forecasted weather for the next five days. The scope of the information displayed is limited to showing the minimum and maximum temperature, pressure, humidity, visibility and cloud percentage. The temperature is only displayed in the form of Celsius.

Detailed working of the project:

A number of packages have been imported to use the functions provided by the different classes. The io package has been imported to utilize the FileWriter and BufferedReader classes. The net package has been imported to utilize the URL class, to create a URL object to hold an URL, as well as the HttpURLConnection class to establish an API connection. The gson package provided by Google has been used to work with JSON files. The swing package has been used to create the application, and the awt package has been imported to use the Font class.

All functionality is occurring within the WeatherForecaster class. It contains two functions – the user-defined APIRequest() and the main() function.
The APIRequest() function has been created to establish a link with the OpenWeatherMap servers, to get current weather data, and to save the JSON data within a file.
The URL to the API has been saved within

the URL object weatherAPI. A connection has been opened to it using the openConnection() function, which has been type casted into a HttpURLConnection object, so that the setRequestMethod can be used. The openConnection() function returns a URL instance, but an HttpURLConnection instance is needed, hence the need for the typecasting.
The request sent to the API is in the form of the GET method, to receive data from the server.
An instance of the BufferedReader class is used to buffer the input stream that would be coming in from the server. The data from the server is taken in using getInputStream(). The FileWriter class instanance 'fw' has been used to write the data received from the server, into the mumbai.json file.
At the end, the FileWriter instance has been closed, along with the BufferedReader instance and the API conncetion.

In the main() function -
The APIRequest() function has then been called, followed by the data being read in from the World Weather Information Service website, to store the forecast data.
A BufferedReader and a FileWriter instance have been used to read and store the data. The openStream() function has been used to read data from the URL.
After writing the data to the mumbaiforecast.json file, the FileWriter and BufferedReader instances have been closed.

A JFrame has been created to hold all the data. A JTabbedPane instance has been created to create tabs to switch between a panel displaying the current weather information and the forecasted data.
The two panels created are 'today' and 'forecasts', respectively. The layout manager for both the panels has been set to null.

As the JsonParser class of the gson package has a function that requires a Reader class instance to parse the JSON data (JsonParser.parseReader(), where the instance of the Reader is passed as a parameter), two Reader instances have been opened on the two json files- 'mumbai.json' and 'mumbaiforecast.json'.

JsonObject instances have been created to get one JSON object literal, that is, a key-value pair.
Within a JSON object, multiple JSON objects can exist. The JsonObject class has the getAsJsonObject() method, where we can pass the key, to get the corresponding value. The value of a key can be either a string, a JSON object or a JSON array.
Each corresponding value that has been

printed on the panel has a corresponding label. The values taken from the keys have been converted into a string, using the getAsString() function of the JsonObject class.
The labels will display the corresponding string values.
The setFont() function has been used to set the font style, whether or not it should be displayed plainly or in bold, as well as to set the font size.
The setBounds() function is used to set the x-axis coordinate, y-axis coordinate, width and height, which would be passed in this order, as parameters to the function.
The add() function is used to add the label to the panel.

Data from the JSON array can be extracted using the getAsJsonArray() method, where the key containing the array is passed as a parameter. Different values of the array can be accessed using the get() method, where the index value of the array value is passed.

Additionally, as the current weather data also has descriptions of the weather as well, a corresponding icon can be displayed as well. This can be achieved using the ImageIcon class, which is then is then converted into a JLabel, and placed accordingly.

Similarly, according to the cloud percentage, the icon can be changed as well.

The background color of the today panel has been set using the setBackground() method, and the tabs have been added to the JTabbedPane using the addTab() method.

A logo for the application has been set using the setIconImage() method, where the .png file has been passed as a parameter. The image can be extracted using the getImage() method.

The getContentPane() function is used to retrieve the content pane layer, so that the JTabbedPane can be added to the frame.

The setExtendedState(JFrame.MAXIMIZED_BOTH) function is used to fully maximize the window vertically and horizontally.

The setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE) function is used to close the window on pressing the ‘X” button.

The setVisible() function has the ‘true’ parameter passed to it, to display the frame.

Both the functions throw Exceptions, owing to the fact that the BufferedReader and the FileWriter instances are used, thus

both functions mention ‘throws Exception’ in the function header definition.

Use and Purpose:

The application can be used to display current weather information and forecast data.

Merits and Demerits:

A merit is that the data displayed can be parsed through quickly, and it is refreshed every time the program is executed. However, the data is not displayed live, which means that any changes in the data would only be reflected when the app is closed and opened once again.

Future Enhancements:

A major area to improve would be to display the data live. Aiming to predict the weather using a machine learning model would be an interesting avenue to pursue.

References:

Web References:

https://worldweather.wmo.int/en/dataguide.html

https://www.youtube.com/watch?v=o_MouGNA1AE&ab_channel=FredOverflow

https://www.programiz.com/java-programming/reader

https://stackoverflow.com/questions/60771386/jsonparser-is-deprecated

https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#:~:text=public%20class%20BufferedReader%20extends%20Reader,large%20enough%20for%20most%20purposes.

https://docs.oracle.com/javase/tutorial/essential/io/buffers.html

https://www.javatpoint.com/java-get-post

https://docs.oracle.com/javase/tutorial/networking/urls/readingURL.html

https://docs.oracle.com/javase/8/docs/api/java/io/InputStreamReader.html

https://www.sarthaks.com/3503147/how-do-i-set-the-font-size-of-a-jlabel-in-java

https://stackoverflow.com/questions/1081486/setting-background-color-for-a-jframe

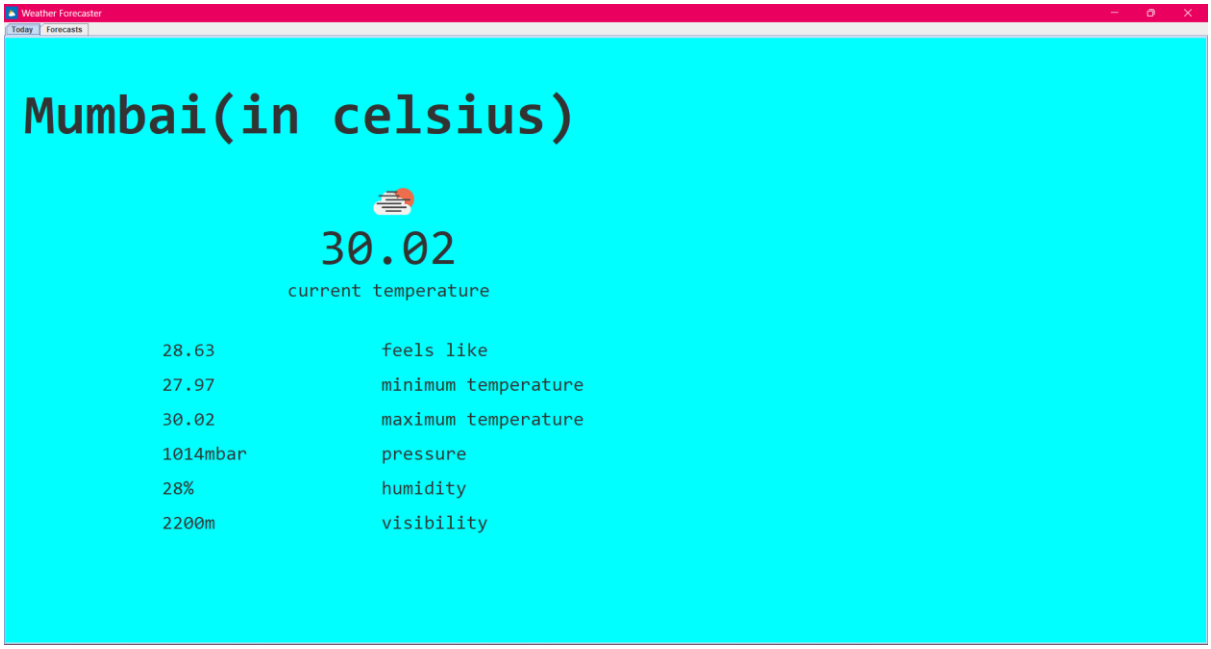
https://medium.com/@michael71314/java-lesson-22-inserting-images-onto-the-jframe-a0a0b6540cca

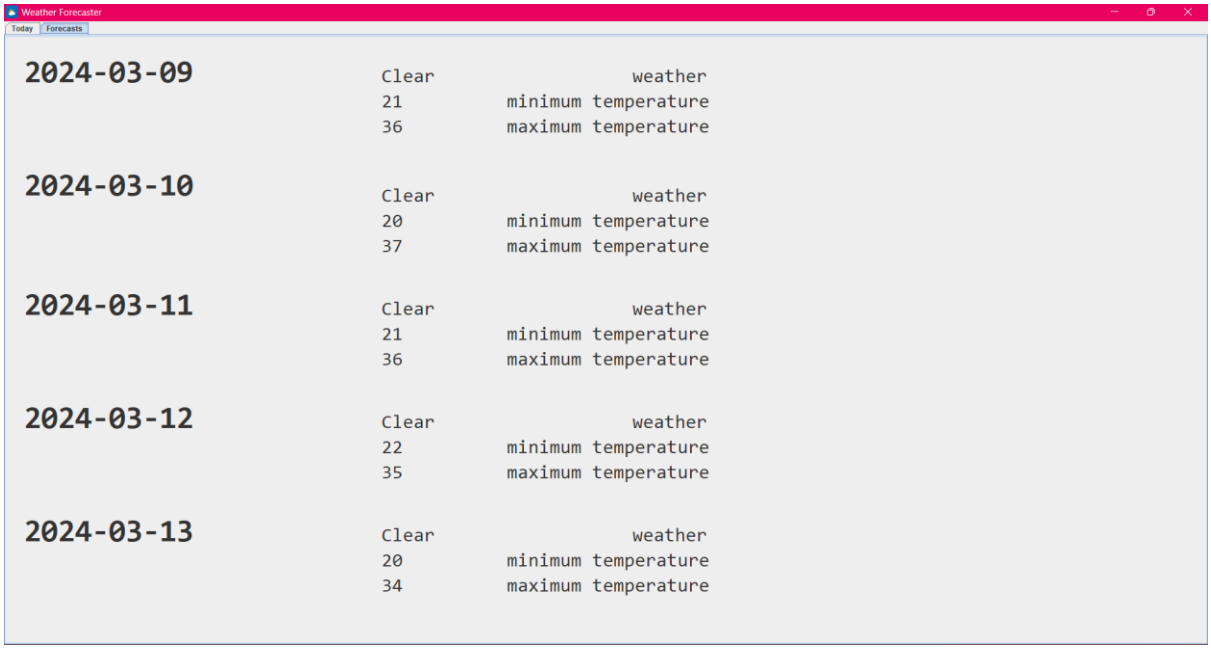
https://openweathermap.org/weather-conditions

https://openweathermap.org/current#multi

https://www.oreilly.com/library/view/learning-java-4th/9781449372477/ch17s10.html

Screenshots:





Source Code:

```
import java.io.*;

import java.net.*;

import com.google.gson.*;

import javax.swing.*;

import java.awt.*;


class WeatherForecaster

{

	public static void APIRequest() throws Exception

	{

		URL weatherAPI = new
URL("https://api.openweathermap.org/data/2.5/weather?lat=19.07&lon=72.87&appid=59b85
a4b5ef2d64564298deb0f11d0db&units=metric");

		HttpURLConnection apiConnection = (HttpURLConnection)
weatherAPI.openConnection();

		apiConnection.setRequestMethod("GET");

		BufferedReader in = new BufferedReader(new
InputStreamReader(apiConnection.getInputStream()));

		FileWriter fw = new FileWriter("mumbai.json");

		String inputPaste;

		while((inputPaste=in.readLine())!=null)

			fw.write(inputPaste);

		fw.close();

		in.close();

		apiConnection.disconnect();
```

```
            //System.out.println("Data saved in mumbai.json");
        }

        public static void main(String args[]) throws Exception
        {
            APIRequest();
            URL weatherData = new
URL("https://worldweather.wmo.int/en/json/226_en.json");
            BufferedReader in = new BufferedReader(new
InputStreamReader(weatherData.openStream()));
            FileWriter fw = new FileWriter("mumbaiforecast.json");

            String inputPaste;
            while((inputPaste=in.readLine())!=null)
                fw.write(inputPaste);
            fw.close();
            in.close();
            //System.out.println("Data saved in mumbaiforecast.json");

            JFrame frame = new JFrame("Weather Forecaster");
            //frame.setLayout(null);
            JTabbedPane weatherForecasts = new JTabbedPane();

            JPanel today = new JPanel();
            today.setLayout(null);
```

```
Reader w = new FileReader("mumbai.json");
Reader wf = new FileReader("mumbaiforecast.json");
JsonObject w_full = JsonParser.parseReader(w).getAsJsonObject();
JsonObject wf_full = JsonParser.parseReader(wf).getAsJsonObject();

JsonObject jobject1 = wf_full.getAsJsonObject("city");
String city = jobject1.get("cityName").getAsString();
JLabel city_label = new JLabel(city + "(in celsius)");
city_label.setFont(new Font("Consolas", Font.BOLD, 90));
city_label.setBounds(30,80,900,100);
today.add(city_label);

JsonArray weatherArray = w_full.getAsJsonArray("weather");
JsonObject weatherDescription = weatherArray.get(0).getAsJsonObject();
String condition = weatherDescription.get("main").getAsString();
ImageIcon cloudsIcon;
JLabel cloudsIcon_label;
//System.out.println(condition);
switch(condition)
{
        case "Thunderstorm":
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini
Project/weather icons/thunderstorm.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
```

```
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
                break;

            case "Drizzle":
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini Project/weather icons/showerrain.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
                break;

            case "Rain":
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini Project/weather icons/rain.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
                break;

            case "Smoke":
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini Project/weather icons/mist.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
```

```
                today.add(cloudsIcon_label);
                break;

            case "Clear":
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini Project/weather icons/clearsky.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
                break;
        }
        JsonObject cloudsDescription = w_full.getAsJsonObject("clouds");
        double cloudsPercentage = cloudsDescription.get("all").getAsDouble();
        //System.out.println(cloudsPercentage);
        if(cloudsPercentage>11.00 && cloudsPercentage<=25.00)
        {
            cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini Project/weather icons/fewclouds.png");
            cloudsIcon_label = new JLabel(cloudsIcon);
            cloudsIcon_label.setBounds(569,210,100,100);
            today.add(cloudsIcon_label);
        }
        else if(cloudsPercentage>25.00 && cloudsPercentage<=50.00)
        {
```

```
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini
Project/weather icons/scatteredclouds.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
            }
            else if(cloudsPercentage>=51.00 && cloudsPercentage<=100.00)
            {
                cloudsIcon = new ImageIcon("D:/Studies/Core Java Mini
Project/weather icons/brokenclouds.png");
                cloudsIcon_label = new JLabel(cloudsIcon);
                cloudsIcon_label.setBounds(569,210,100,100);
                today.add(cloudsIcon_label);
            }

            JsonObject jobject2 = w_full.getAsJsonObject("main");
            String temp = jobject2.get("temp").getAsString();
            JLabel temp_label = new JLabel(temp);
            temp_label.setFont(new Font("Consolas", Font.PLAIN, 80));
            temp_label.setBounds(500,290,2000,100);
            today.add(temp_label);

            JLabel tempDenote_label = new JLabel("current temperature");
            tempDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
            tempDenote_label.setBounds(450,390,500,30);
```

```
today.add(tempDenote_label);

String feelsLike = jobject2.get("feels_like").getAsString();
JLabel feelsLike_label = new JLabel(feelsLike);
feelsLike_label.setFont(new Font("Consolas", Font.PLAIN, 30));
feelsLike_label.setBounds(250,485,150,30);
today.add(feelsLike_label);

JLabel feelsLikeDenote_label = new JLabel("feels like");
feelsLikeDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
feelsLikeDenote_label.setBounds(600,485,500,30);
today.add(feelsLikeDenote_label);

String minTemp = jobject2.get("temp_min").getAsString();
JLabel minTemp_label = new JLabel(minTemp);
minTemp_label.setFont(new Font("Consolas", Font.PLAIN, 30));
minTemp_label.setBounds(250,540,100,30);
today.add(minTemp_label);

JLabel minTempDenote_label = new JLabel("minimum temperature");
minTempDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
minTempDenote_label.setBounds(600,540,500,30);
today.add(minTempDenote_label);
```

```
String maxTemp = jobject2.get("temp_max").getAsString();
JLabel maxTemp_label = new JLabel(maxTemp);
maxTemp_label.setFont(new Font("Consolas", Font.PLAIN, 30));
maxTemp_label.setBounds(250,595,100,30);
today.add(maxTemp_label);

JLabel maxTempDenote_label = new JLabel("maximum temperature");
maxTempDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
maxTempDenote_label.setBounds(600,595,500,30);
today.add(maxTempDenote_label);

String pressure = jobject2.get("pressure").getAsString();
JLabel pressure_label = new JLabel(pressure+"mbar");
pressure_label.setFont(new Font("Consolas", Font.PLAIN, 30));
pressure_label.setBounds(250,650,300,30);
today.add(pressure_label);

JLabel pressureDenote_label = new JLabel("pressure");
pressureDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
pressureDenote_label.setBounds(600,650,500,30);
today.add(pressureDenote_label);

String humidity = jobject2.get("humidity").getAsString();
JLabel humidity_label = new JLabel(humidity+"%");
```

```
humidity_label.setFont(new Font("Consolas", Font.PLAIN, 30));
humidity_label.setBounds(250,705,200,30);
today.add(humidity_label);

JLabel humidityDenote_label = new JLabel("humidity");
humidityDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
humidityDenote_label.setBounds(600,705,500,30);
today.add(humidityDenote_label);

String visibility = w_full.get("visibility").getAsString();
JLabel visibility_label = new JLabel(visibility+"m");
visibility_label.setFont(new Font("Consolas", Font.PLAIN, 30));
visibility_label.setBounds(250,760,200,30);
today.add(visibility_label);

JLabel visibilityDenote_label = new JLabel("visibility");
visibilityDenote_label.setFont(new Font("Consolas", Font.PLAIN, 30));
visibilityDenote_label.setBounds(600,760,200,30);
today.add(visibilityDenote_label);

JsonObject forecast = jobject1.getAsJsonObject("forecast");
JsonArray forecastDay = forecast.getAsJsonArray("forecastDay");

JPanel forecasts = new JPanel();
```

```
forecasts.setLayout(null);
JsonObject forecastDay1Json = forecastDay.get(0).getAsJsonObject();
String day1Date = forecastDay1Json.get("forecastDate").getAsString();

JLabel day1DateLabel = new JLabel(day1Date);
day1DateLabel.setFont(new Font("Consolas", Font.BOLD, 50));
day1DateLabel.setBounds(30,10,900,100);
forecasts.add(day1DateLabel);

String day1Weather = forecastDay1Json.get("weather").getAsString();
JLabel day1WeatherLabel = new JLabel(day1Weather);
day1WeatherLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day1WeatherLabel.setBounds(600,50,200,30);
forecasts.add(day1WeatherLabel);

JLabel day1WeatherDescription_label = new JLabel("weather");
day1WeatherDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day1WeatherDescription_label.setBounds(1000,50,200,30);
forecasts.add(day1WeatherDescription_label);

String day1MinTemp = forecastDay1Json.get("minTemp").getAsString();
JLabel day1MinTempLabel = new JLabel(day1MinTemp);
day1MinTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day1MinTempLabel.setBounds(600,90,200,30);
```

```
forecasts.add(day1MinTempLabel);

JLabel day1MinTempDescription_label = new JLabel("minimum
temperature");
day1MinTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day1MinTempDescription_label.setBounds(800,90,500,30);
forecasts.add(day1MinTempDescription_label);

String day1MaxTemp = forecastDay1Json.get("maxTemp").getAsString();
JLabel day1MaxTempLabel = new JLabel(day1MaxTemp);
day1MaxTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day1MaxTempLabel.setBounds(600,130,200,30);
forecasts.add(day1MaxTempLabel);

JLabel day1MaxTempDescription_label = new JLabel("maximum
temperature");
day1MaxTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day1MaxTempDescription_label.setBounds(800,130,500,30);
forecasts.add(day1MaxTempDescription_label);

JsonObject forecastDay2=forecastDay.get(1).getAsJsonObject();
String day2Date=forecastDay2.get("forecastDate").getAsString();
```

```
JLabel day2DateLabel=new JLabel(day2Date);
day2DateLabel.setFont(new Font("Consolas", Font.BOLD, 50));
day2DateLabel.setBounds(30,190,500,100);
forecasts.add(day2DateLabel);

String day2Weather=forecastDay2.get("weather").getAsString();
JLabel day2WeatherLabel = new JLabel(day2Weather);
day2WeatherLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day2WeatherLabel.setBounds(600,240,200,30);
forecasts.add(day2WeatherLabel);

JLabel day2WeatherDescription_label = new JLabel("weather");
day2WeatherDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day2WeatherDescription_label.setBounds(1000,240,200,30);
forecasts.add(day2WeatherDescription_label);

String day2MinTemp=forecastDay2.get("minTemp").getAsString();
JLabel day2MinTempLabel = new JLabel(day2MinTemp);
day2MinTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day2MinTempLabel.setBounds(600,280,200,30);
forecasts.add(day2MinTempLabel);

JLabel day2MinTempDescription_label = new JLabel("minimum
temperature");
```

```
day2MinTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day2MinTempDescription_label.setBounds(800,280,500,30);
forecasts.add(day2MinTempDescription_label);

String day2MaxTemp=forecastDay2.get("maxTemp").getAsString();
JLabel day2MaxTempLabel = new JLabel(day2MaxTemp);
day2MaxTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day2MaxTempLabel.setBounds(600,320,200,30);
forecasts.add(day2MaxTempLabel);

JLabel day2MaxTempDescription_label = new JLabel("maximum
temperature");
day2MaxTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day2MaxTempDescription_label.setBounds(800,320,500,30);
forecasts.add(day2MaxTempDescription_label);

JsonObject forecastDay3=forecastDay.get(2).getAsJsonObject();

String day3Date=forecastDay3.get("forecastDate").getAsString();
JLabel day3DateLabel=new JLabel(day3Date);
day3DateLabel.setFont(new Font("Consolas", Font.BOLD, 50));
day3DateLabel.setBounds(30,380,500,100);
forecasts.add(day3DateLabel);
```

```
                String day3Weather = forecastDay3.get("weather").getAsString();
                JLabel day3WeatherLabel = new JLabel(day3Weather);
                day3WeatherLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
                day3WeatherLabel.setBounds(600,420,500,30);
                forecasts.add(day3WeatherLabel);

                JLabel day3WeatherDescription_label = new JLabel("weather");
                day3WeatherDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
                day3WeatherDescription_label.setBounds(1000,420,500,30);
                forecasts.add(day3WeatherDescription_label);

                String day3MinTemp = forecastDay3.get("minTemp").getAsString();
                JLabel day3MinTempLabel = new JLabel(day3MinTemp);
                day3MinTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
                day3MinTempLabel.setBounds(600,460,500,30);
                forecasts.add(day3MinTempLabel);

                JLabel day3MinTempDescription_label = new JLabel("minimum
temperature");
                day3MinTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
                day3MinTempDescription_label.setBounds(800,460,500,30);
                forecasts.add(day3MinTempDescription_label);
```

```
String day3MaxTemp = forecastDay3.get("maxTemp").getAsString();
JLabel day3MaxTempLabel = new JLabel(day3MaxTemp);
day3MaxTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day3MaxTempLabel.setBounds(600,500,500,30);
forecasts.add(day3MaxTempLabel);

JLabel day3MaxTempDescription_label = new JLabel("maximum
temperature");
day3MaxTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day3MaxTempDescription_label.setBounds(800,500,500,30);
forecasts.add(day3MaxTempDescription_label);

JsonObject forecastDay4 = forecastDay.get(3).getAsJsonObject();

String day4Date = forecastDay4.get("forecastDate").getAsString();
JLabel day4DateLabel = new JLabel(day4Date);
day4DateLabel.setFont(new Font("Consolas", Font.BOLD, 50));
day4DateLabel.setBounds(30,560,500,100);
forecasts.add(day4DateLabel);

String day4Weather = forecastDay4.get("weather").getAsString();
JLabel day4WeatherLabel = new JLabel(day4Weather);
day4WeatherLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
```

```
day4WeatherLabel.setBounds(600,600,500,30);
forecasts.add(day4WeatherLabel);

JLabel day4WeatherDescription_label = new JLabel("weather");
day4WeatherDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day4WeatherDescription_label.setBounds(1000,600,500,30);
forecasts.add(day4WeatherDescription_label);

String day4MinTemp = forecastDay4.get("minTemp").getAsString();
JLabel day4MinTempLabel = new JLabel(day4MinTemp);
day4MinTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day4MinTempLabel.setBounds(600,640,500,30);
forecasts.add(day4MinTempLabel);

JLabel day4MinTempDescription_label = new JLabel("minimum
temperature");
day4MinTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day4MinTempDescription_label.setBounds(800,640,500,30);
forecasts.add(day4MinTempDescription_label);

String day4MaxTemp=forecastDay4.get("maxTemp").getAsString();
JLabel day4MaxTempLabel = new JLabel(day4MaxTemp);
day4MaxTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
```

```
day4MaxTempLabel.setBounds(600,680,500,30);
forecasts.add(day4MaxTempLabel);

JLabel day4MaxTempDescription_label = new JLabel("maximum
temperature");
day4MaxTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day4MaxTempDescription_label.setBounds(800,680,500,30);
forecasts.add(day4MaxTempDescription_label);

JsonObject forecastDay5 = forecastDay.get(4).getAsJsonObject();

String day5Date = forecastDay5.get("forecastDate").getAsString();
JLabel day5DateLabel = new JLabel(day5Date);
day5DateLabel.setFont(new Font("Consolas", Font.BOLD, 50));
day5DateLabel.setBounds(30,740,500,100);
forecasts.add(day5DateLabel);

String day5Weather = forecastDay5.get("weather").getAsString();
JLabel day5WeatherLabel = new JLabel(day5Weather);
day5WeatherLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day5WeatherLabel.setBounds(600,780,500,30);
forecasts.add(day5WeatherLabel);

JLabel day5WeatherDescription_label = new JLabel("weather");
```

```
day5WeatherDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day5WeatherDescription_label.setBounds(1000,780,500,30);
forecasts.add(day5WeatherDescription_label);

String day5MinTemp = forecastDay5.get("minTemp").getAsString();
JLabel day5MinTempLabel = new JLabel(day5MinTemp);
day5MinTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day5MinTempLabel.setBounds(600,820,500,30);
forecasts.add(day5MinTempLabel);

JLabel day5MinTempDescription_label = new JLabel("minimum
temperature");
day5MinTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
day5MinTempDescription_label.setBounds(800,820,500,30);
forecasts.add(day5MinTempDescription_label);

String day5MaxTemp = forecastDay5.get("maxTemp").getAsString();
JLabel day5MaxTempLabel = new JLabel(day5MaxTemp);
day5MaxTempLabel.setFont(new Font("Consolas", Font.PLAIN, 30));
day5MaxTempLabel.setBounds(600,860,500,30);
forecasts.add(day5MaxTempLabel);
```

```
                JLabel day5MaxTempDescription_label = new JLabel("maximum
temperature");
                day5MaxTempDescription_label.setFont(new Font("Consolas", Font.PLAIN,
30));
                day5MaxTempDescription_label.setBounds(800,860,500,30);
                forecasts.add(day5MaxTempDescription_label);


                today.setBackground(Color.cyan);


                weatherForecasts.addTab("Today",today);
                weatherForecasts.addTab("Forecasts",forecasts);


                ImageIcon weatherLogo = new ImageIcon("weatherLogo.png");
                frame.setIconImage(weatherLogo.getImage());


                frame.getContentPane().add(weatherForecasts);
                frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setVisible(true);
        }
}
```