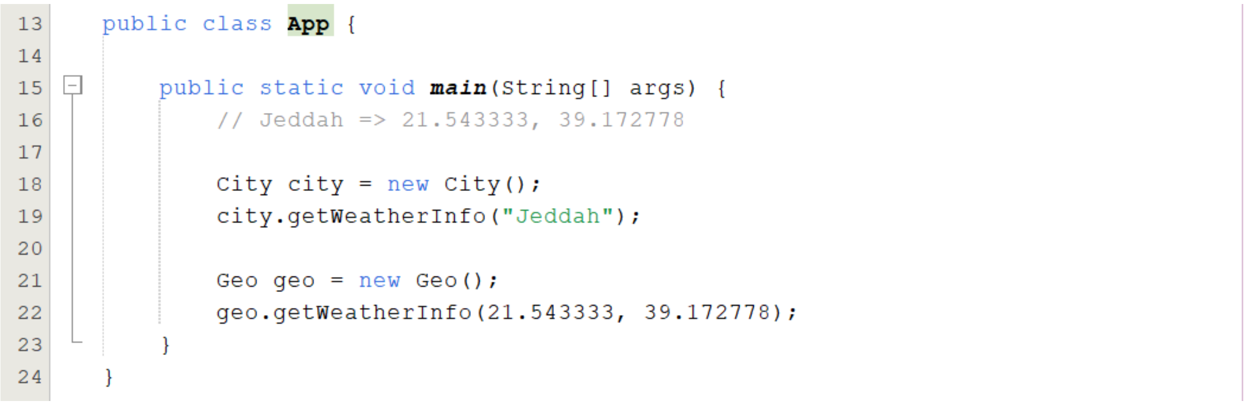
CPIT252

Lab Activity #06

Elaf Yousef Aloufi, 1911265 - CAR

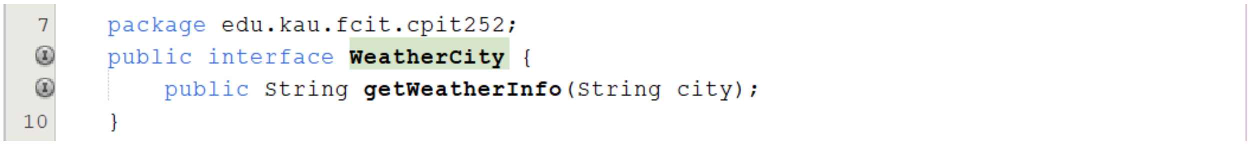
Code

App (main)



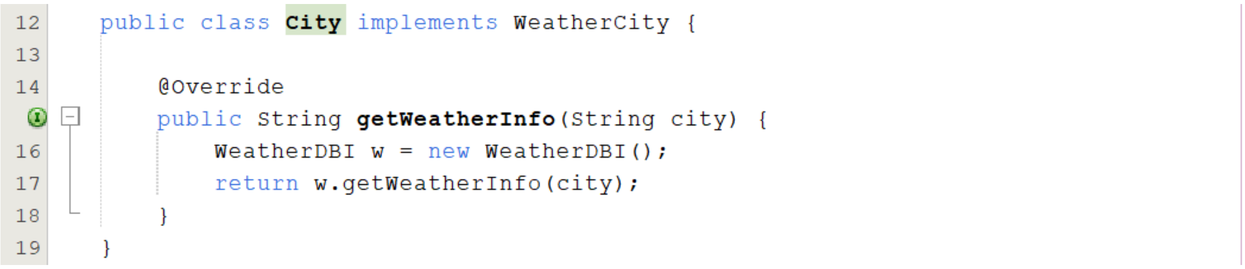
```java
public class App {

    public static void main(String[] args) {
        // Jeddah => 21.543333, 39.172778

        City city = new City();
        city.getWeatherInfo("Jeddah");

        Geo geo = new Geo();
        geo.getWeatherInfo(21.543333, 39.172778);
    }
}
```

WeatherCity Interface



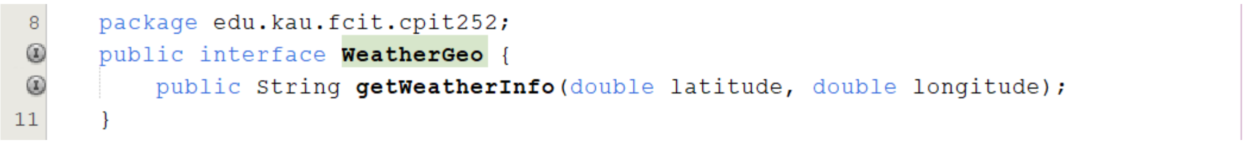
```java
package edu.kau.fcit.cpit252;
public interface WeatherCity {
    public String getWeatherInfo(String city);
}
```

City Class



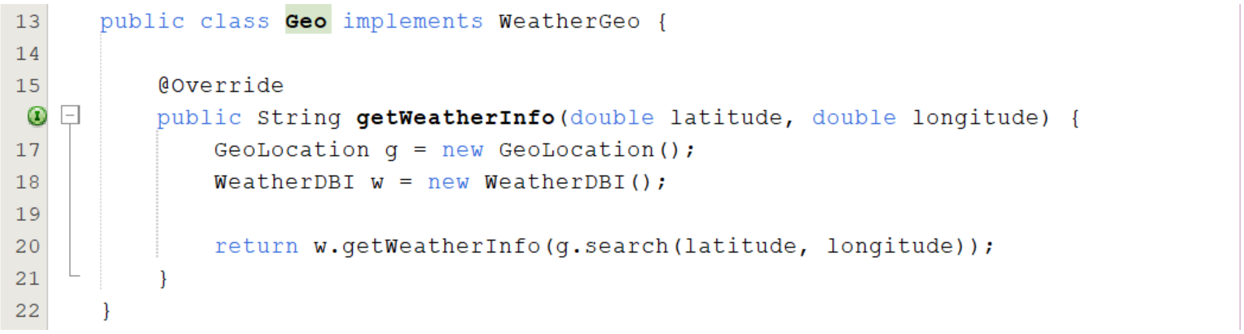
```java
public class City implements WeatherCity {

    @Override
    public String getWeatherInfo(String city) {
        WeatherDBI w = new WeatherDBI();
        return w.getWeatherInfo(city);
    }
}
```

WeatherGeo Interface



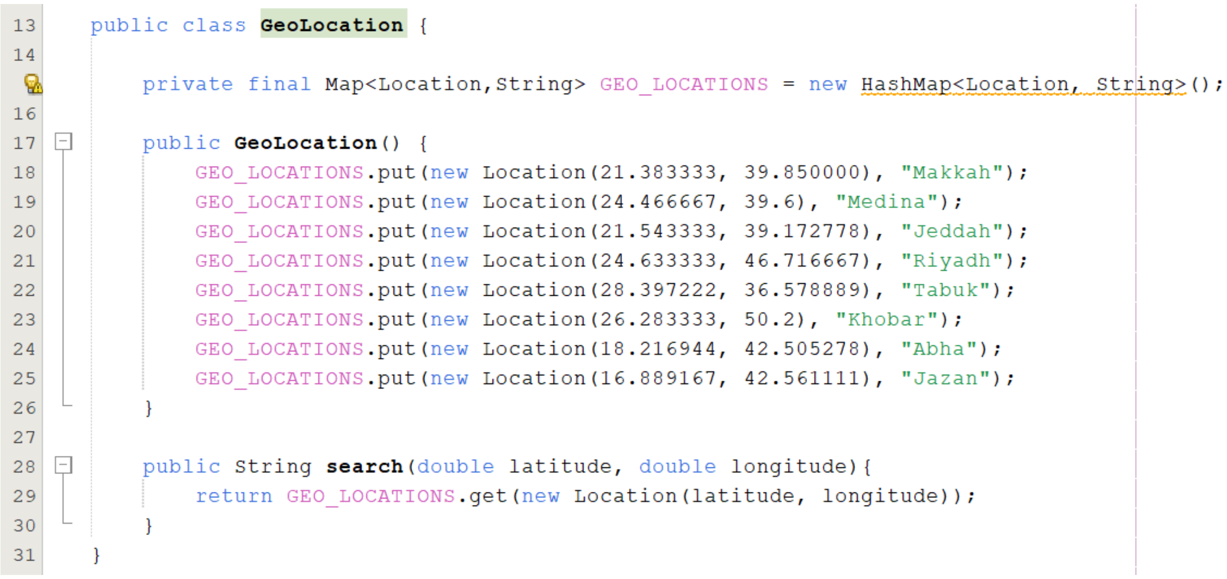
```java
package edu.kau.fcit.cpit252;
public interface WeatherGeo {
    public String getWeatherInfo(double latitude, double longitude);
}
```

Geo Class



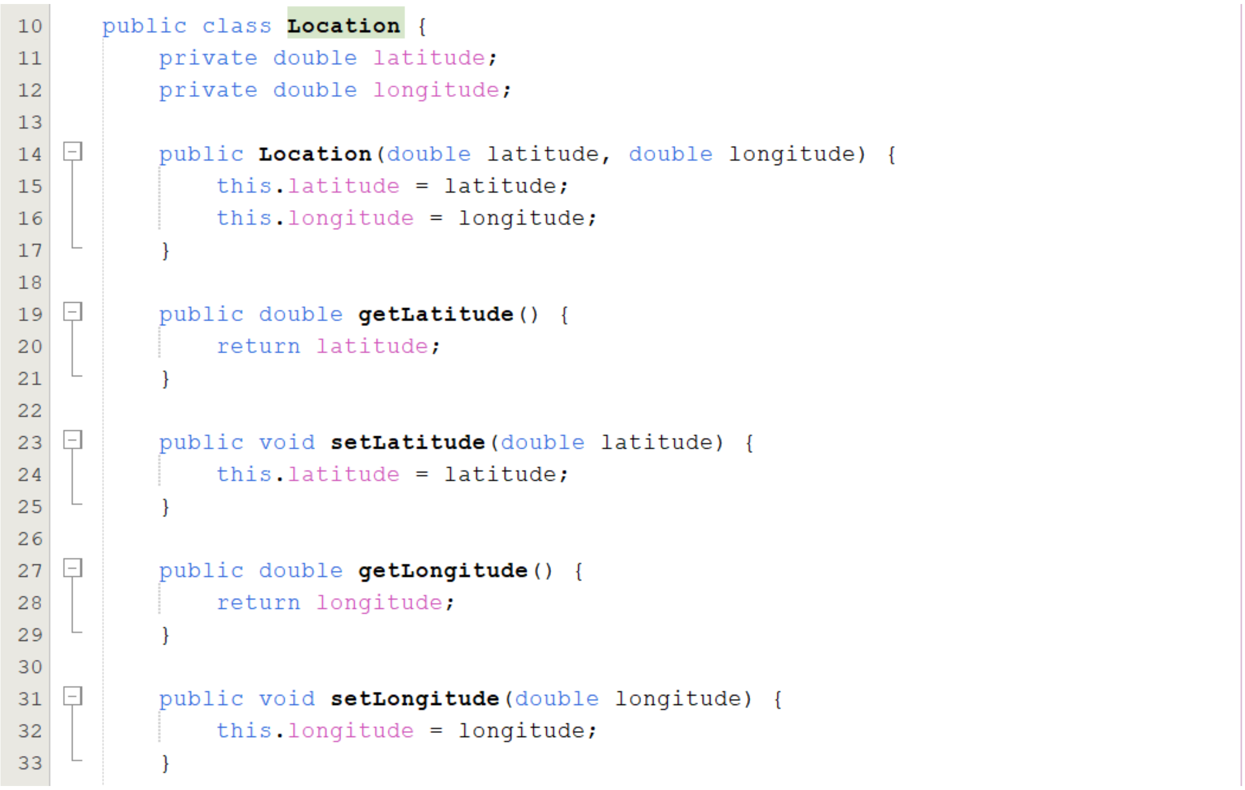
```java
public class Geo implements WeatherGeo {

    @Override
    public String getWeatherInfo(double latitude, double longitude) {
        GeoLocation g = new GeoLocation();
        WeatherDBI w = new WeatherDBI();

        return w.getWeatherInfo(g.search(latitude, longitude));
    }
}
```

GeoLocation Class

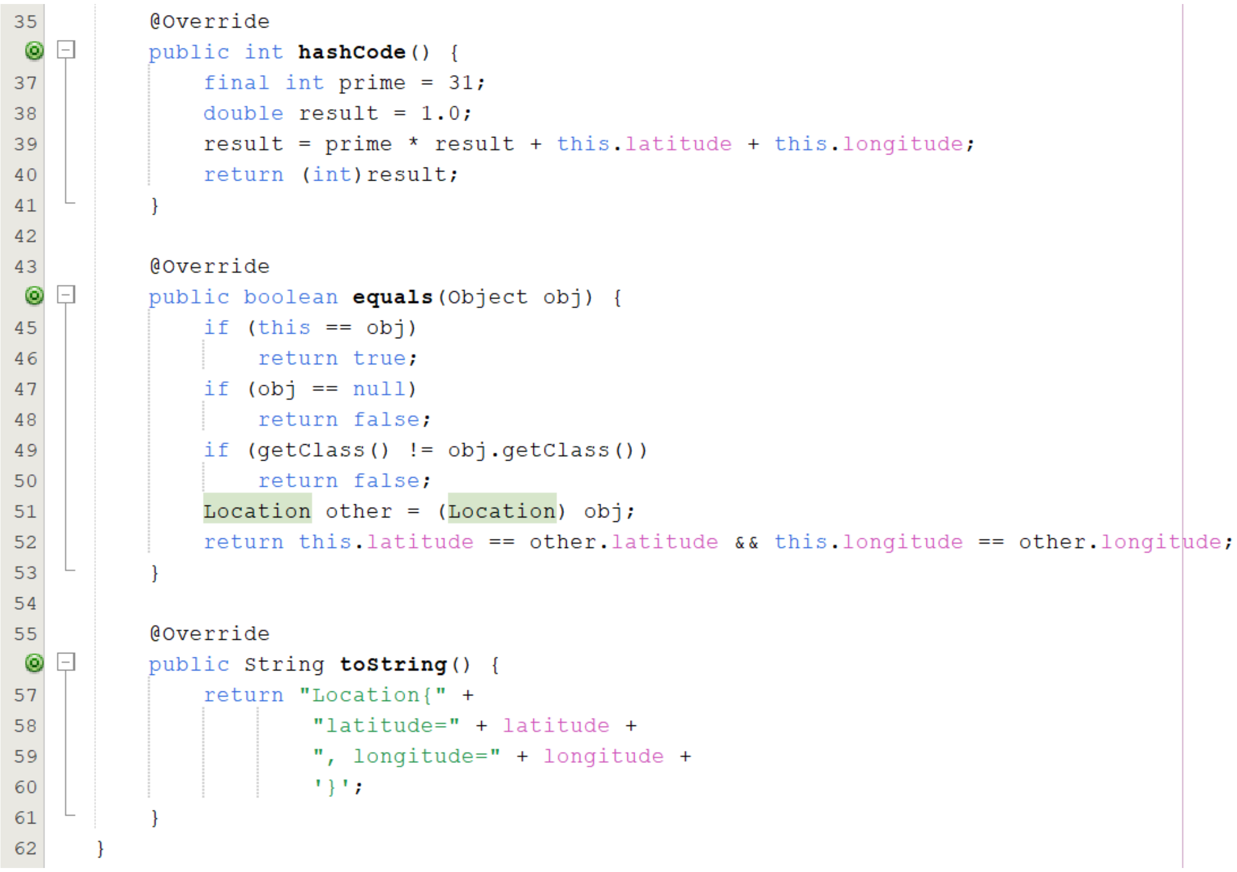


```java
public class GeoLocation {

    private final Map<Location,String> GEO_LOCATIONS = new HashMap<Location, String>();

    public GeoLocation() {
        GEO_LOCATIONS.put(new Location(21.383333, 39.850000), "Makkah");
        GEO_LOCATIONS.put(new Location(24.466667, 39.6), "Medina");
        GEO_LOCATIONS.put(new Location(21.543333, 39.172778), "Jeddah");
        GEO_LOCATIONS.put(new Location(24.633333, 46.716667), "Riyadh");
        GEO_LOCATIONS.put(new Location(28.397222, 36.578889), "Tabuk");
        GEO_LOCATIONS.put(new Location(26.283333, 50.2), "Khobar");
        GEO_LOCATIONS.put(new Location(18.216944, 42.505278), "Abha");
        GEO_LOCATIONS.put(new Location(16.889167, 42.561111), "Jazan");
    }

    public String search(double latitude, double longitude){
        return GEO_LOCATIONS.get(new Location(latitude, longitude));
    }
}
```

Location Class

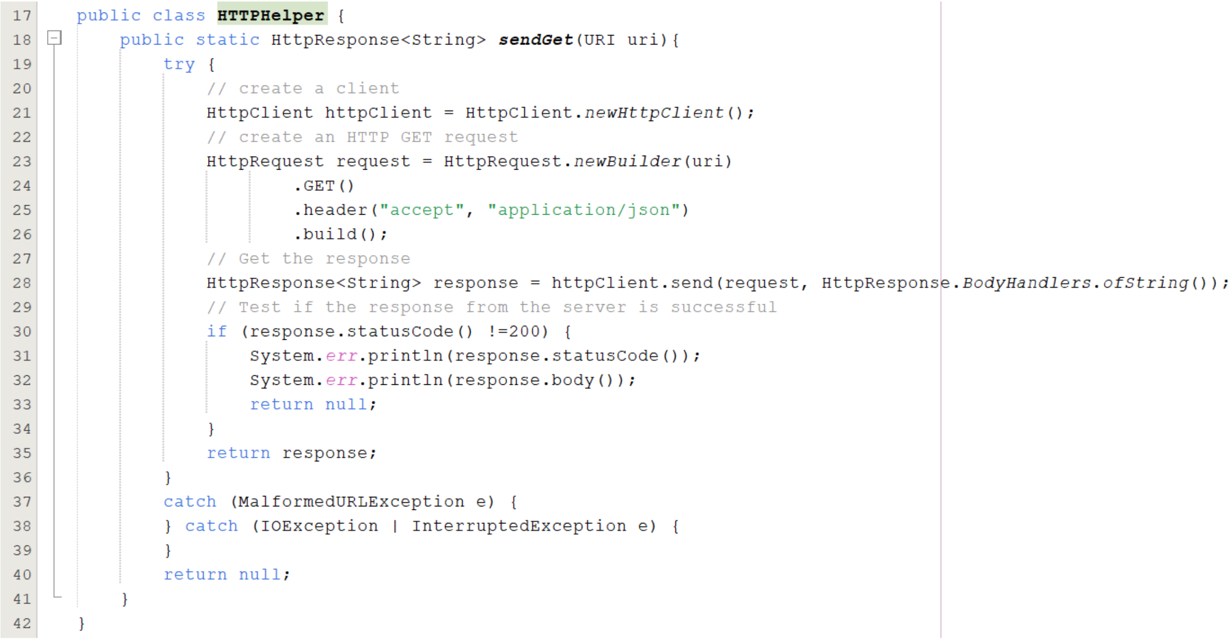


```java
public class Location {
    private double latitude;
    private double longitude;

    public Location(double latitude, double longitude) {
        this.latitude = latitude;
        this.longitude = longitude;
    }

    public double getLatitude() {
        return latitude;
    }

    public void setLatitude(double latitude) {
        this.latitude = latitude;
    }

    public double getLongitude() {
        return longitude;
    }

    public void setLongitude(double longitude) {
        this.longitude = longitude;
    }
```



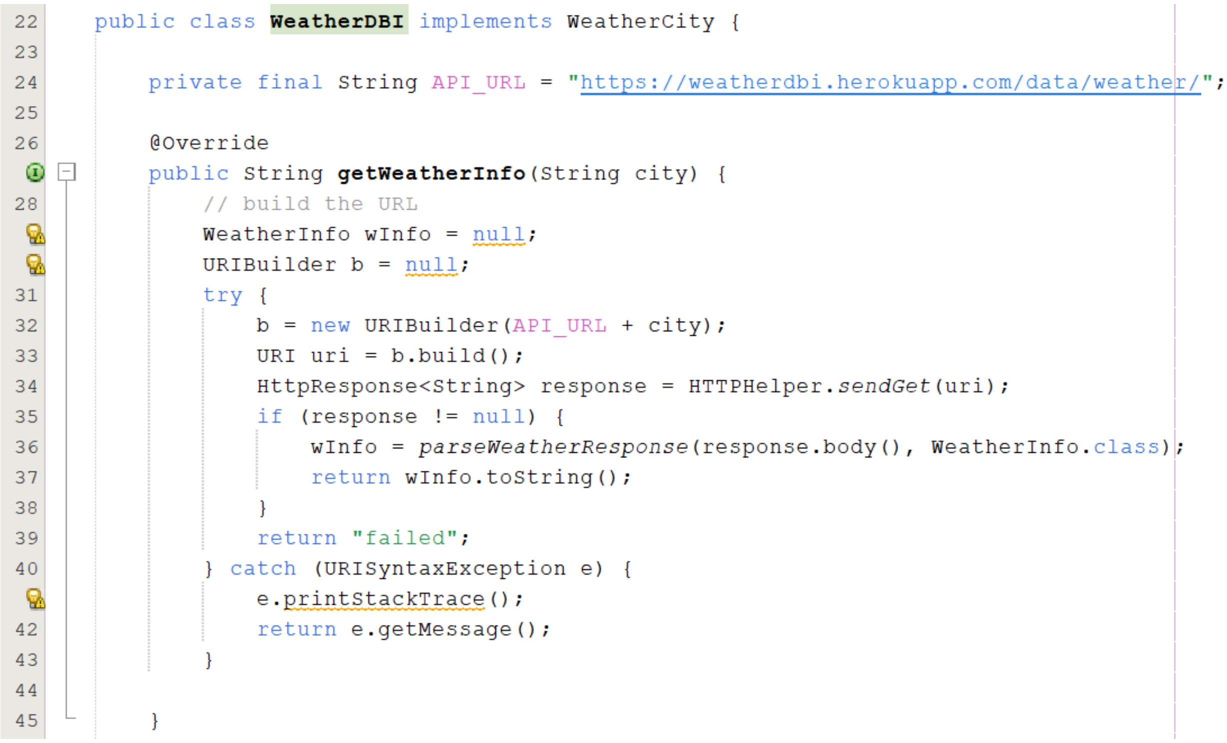
```
    @Override
    public int hashCode() {
        final int prime = 31;
        double result = 1.0;
        result = prime * result + this.latitude + this.longitude;
        return (int)result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Location other = (Location) obj;
        return this.latitude == other.latitude && this.longitude == other.longitude;
    }

    @Override
    public String toString() {
        return "Location{" +
                "latitude=" + latitude +
                ", longitude=" + longitude +
                '}';
    }
}
```

HTTPHelper Class

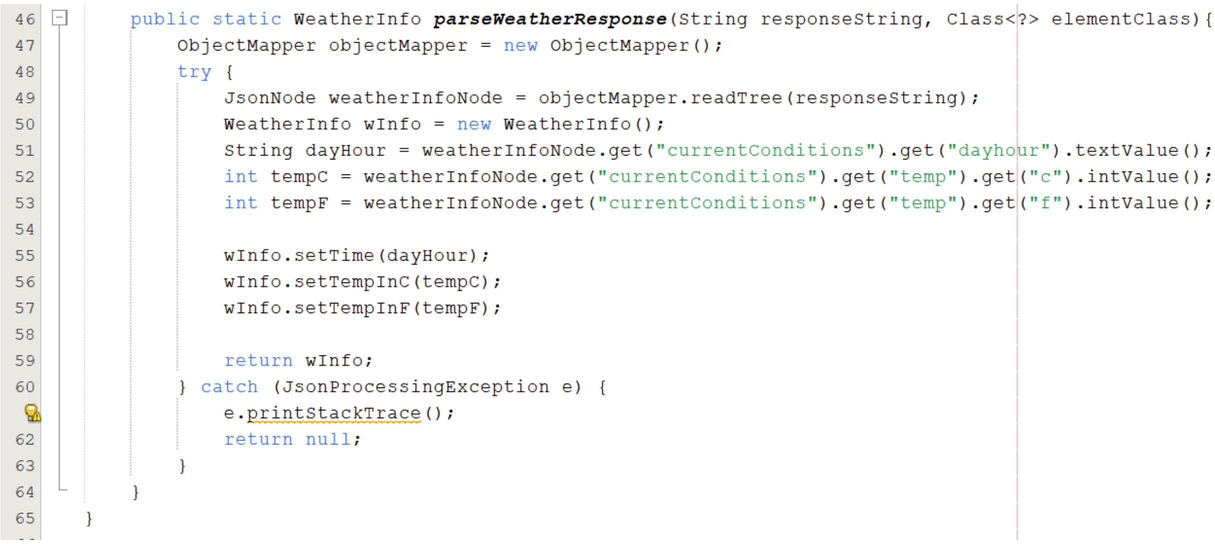


```java
public class HTTPHelper {
    public static HttpResponse<String> sendGet(URI uri){
        try {
            // create a client
            HttpClient httpClient = HttpClient.newHttpClient();
            // create an HTTP GET request
            HttpRequest request = HttpRequest.newBuilder(uri)
                    .GET()
                    .header("accept", "application/json")
                    .build();
            // Get the response
            HttpResponse<String> response = httpClient.send(request, HttpResponse.BodyHandlers.ofString());
            // Test if the response from the server is successful
            if (response.statusCode() !=200) {
                System.err.println(response.statusCode());
                System.err.println(response.body());
                return null;
            }
            return response;
        }
        catch (MalformedURLException e) {
        } catch (IOException | InterruptedException e) {
        }
        return null;
    }
}
```

WeatherDBI Class

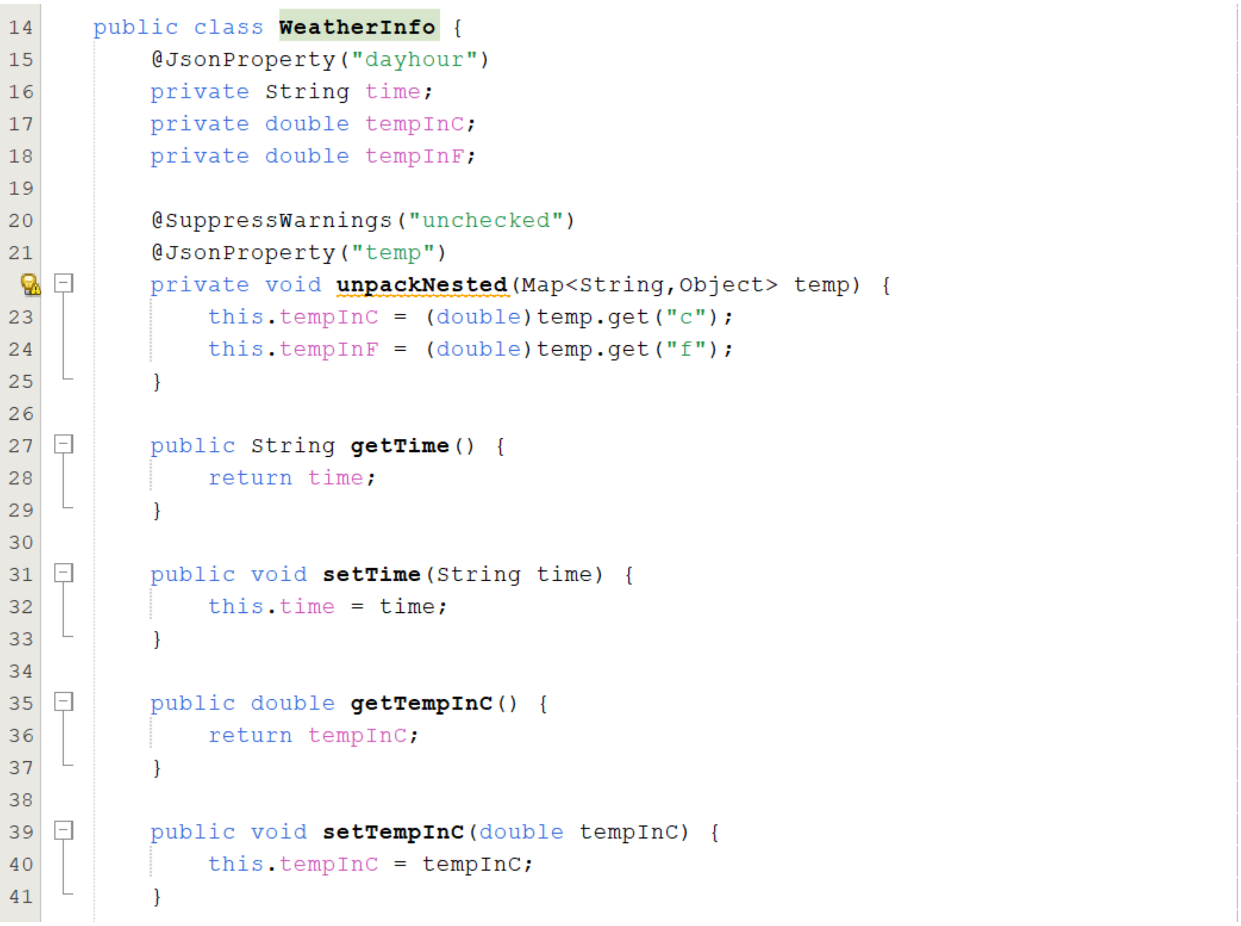


```java
public class WeatherDBI implements WeatherCity {

    private final String API_URL = "https://weatherdbi.herokuapp.com/data/weather/";

    @Override
    public String getWeatherInfo(String city) {
        // build the URL
        WeatherInfo wInfo = null;
        URIBuilder b = null;
        try {
            b = new URIBuilder(API_URL + city);
            URI uri = b.build();
            HttpResponse<String> response = HTTPHelper.sendGet(uri);
            if (response != null) {
                wInfo = parseWeatherResponse(response.body(), WeatherInfo.class);
                return wInfo.toString();
            }
            return "failed";
        } catch (URISyntaxException e) {
            e.printStackTrace();
            return e.getMessage();
        }

    }
```

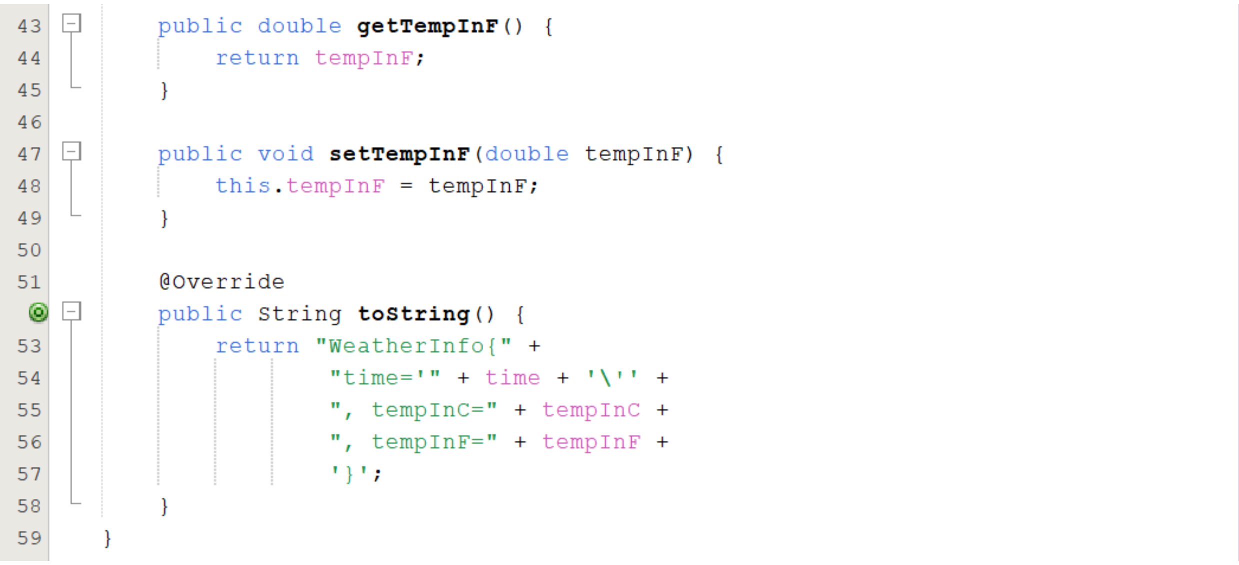


```java
    public static WeatherInfo parseWeatherResponse(String responseString, Class<?> elementClass){
        ObjectMapper objectMapper = new ObjectMapper();
        try {
            JsonNode weatherInfoNode = objectMapper.readTree(responseString);
            WeatherInfo wInfo = new WeatherInfo();
            String dayHour = weatherInfoNode.get("currentConditions").get("dayhour").textValue();
            int tempC = weatherInfoNode.get("currentConditions").get("temp").get("c").intValue();
            int tempF = weatherInfoNode.get("currentConditions").get("temp").get("f").intValue();

            wInfo.setTime(dayHour);
            wInfo.setTempInC(tempC);
            wInfo.setTempInF(tempF);

            return wInfo;
        } catch (JsonProcessingException e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

WeatherInfo Class

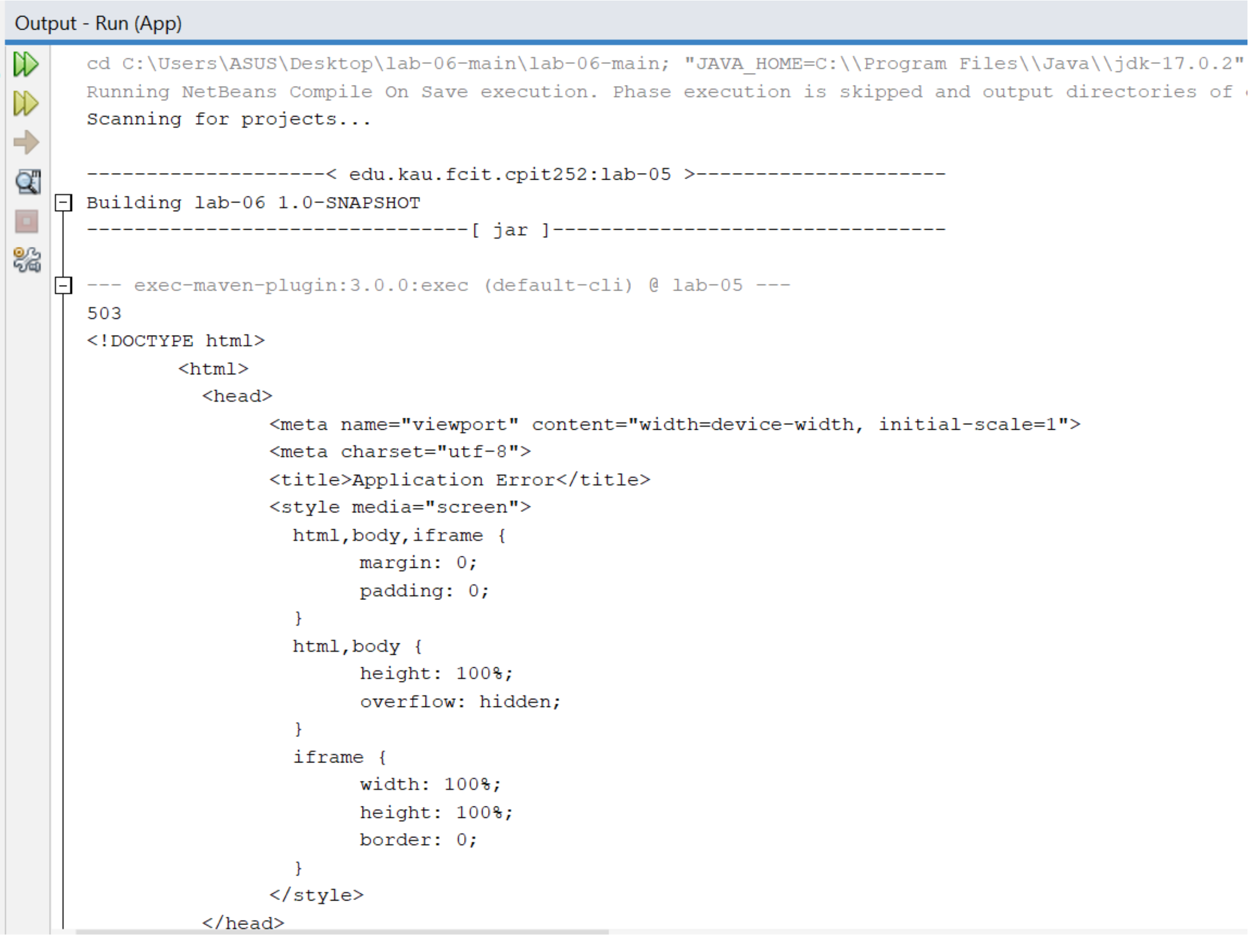


```java
public class WeatherInfo {
    @JsonProperty("dayhour")
    private String time;
    private double tempInC;
    private double tempInF;

    @SuppressWarnings("unchecked")
    @JsonProperty("temp")
    private void unpackNested(Map<String,Object> temp) {
        this.tempInC = (double)temp.get("c");
        this.tempInF = (double)temp.get("f");
    }

    public String getTime() {
        return time;
    }

    public void setTime(String time) {
        this.time = time;
    }

    public double getTempInC() {
        return tempInC;
    }

    public void setTempInC(double tempInC) {
        this.tempInC = tempInC;
    }
```

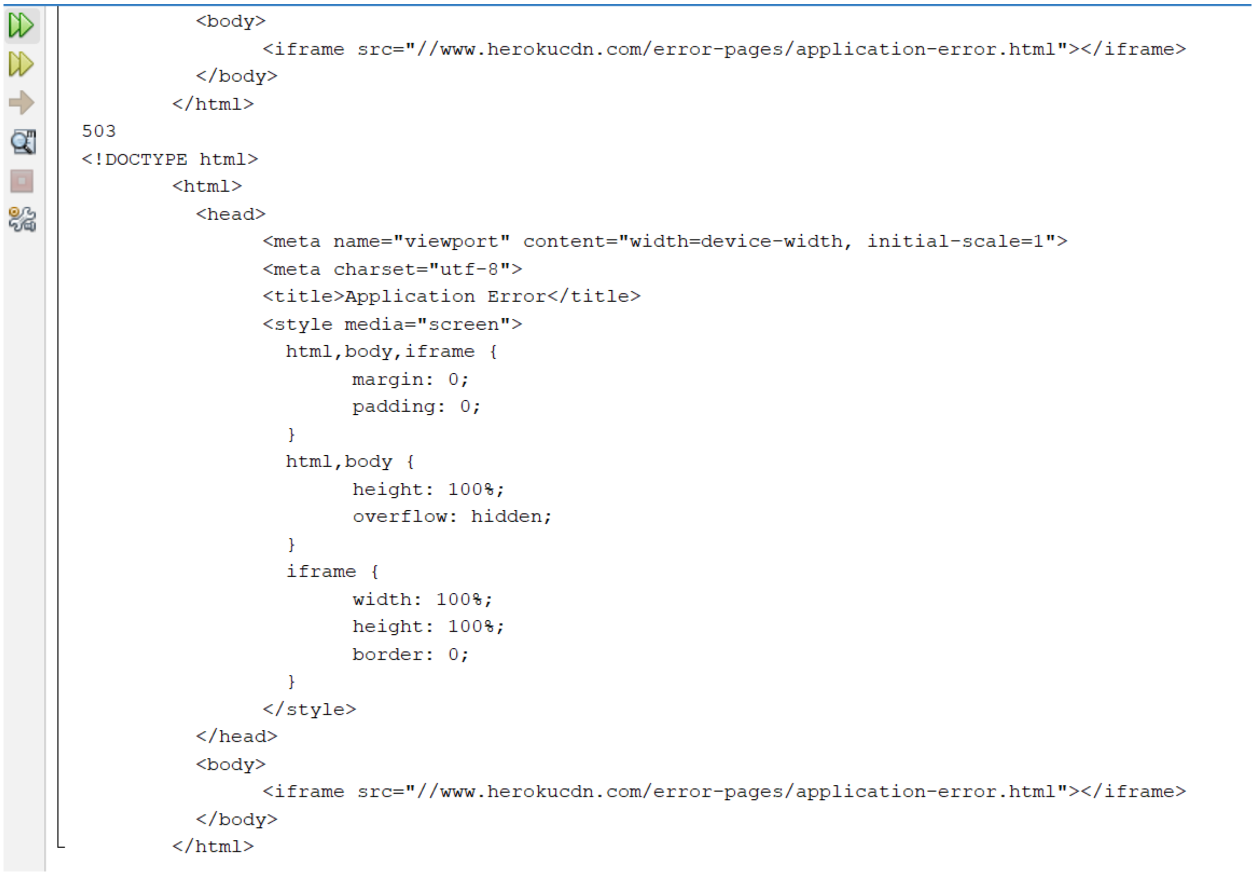


```java
    public double getTempInF() {
        return tempInF;
    }

    public void setTempInF(double tempInF) {
        this.tempInF = tempInF;
    }

    @Override
    public String toString() {
        return "WeatherInfo{" +
                "time='" + time + '\'' +
                ", tempInC=" + tempInC +
                ", tempInF=" + tempInF +
                '}';
    }
}
```

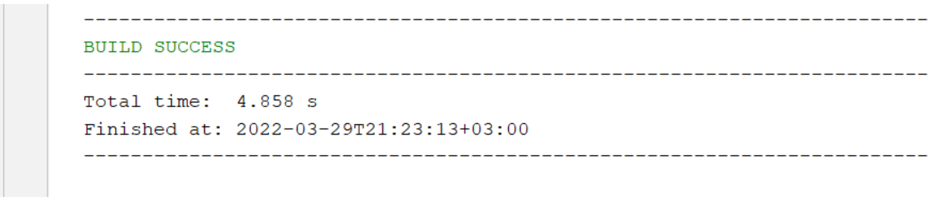
Output



```
cd C:\Users\ASUS\Desktop\lab-06-main\lab-06-main; "JAVA_HOME=C:\\Program Files\\Java\\jdk-17.0.2"
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of
Scanning for projects...

--------------------< edu.kau.fcit.cpit252:lab-05 >---------------------
Building lab-06 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ lab-05 ---
503
<!DOCTYPE html>
        <html>
          <head>
               <meta name="viewport" content="width=device-width, initial-scale=1">
               <meta charset="utf-8">
               <title>Application Error</title>
               <style media="screen">
                 html,body,iframe {
                       margin: 0;
                       padding: 0;
                 }
                 html,body {
                       height: 100%;
                       overflow: hidden;
                 }
                 iframe {
                       width: 100%;
                       height: 100%;
                       border: 0;
                 }
               </style>
          </head>
```



```
          <body>
               <iframe src="//www.herokucdn.com/error-pages/application-error.html"></iframe>
          </body>
        </html>
503
<!DOCTYPE html>
        <html>
          <head>
               <meta name="viewport" content="width=device-width, initial-scale=1">
               <meta charset="utf-8">
               <title>Application Error</title>
               <style media="screen">
                 html,body,iframe {
                       margin: 0;
                       padding: 0;
                 }
                 html,body {
                       height: 100%;
                       overflow: hidden;
                 }
                 iframe {
                       width: 100%;
                       height: 100%;
                       border: 0;
                 }
               </style>
          </head>
          <body>
               <iframe src="//www.herokucdn.com/error-pages/application-error.html"></iframe>
          </body>
        </html>
```



```
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  4.858 s
Finished at: 2022-03-29T21:23:13+03:00
------------------------------------------------------------------------
```