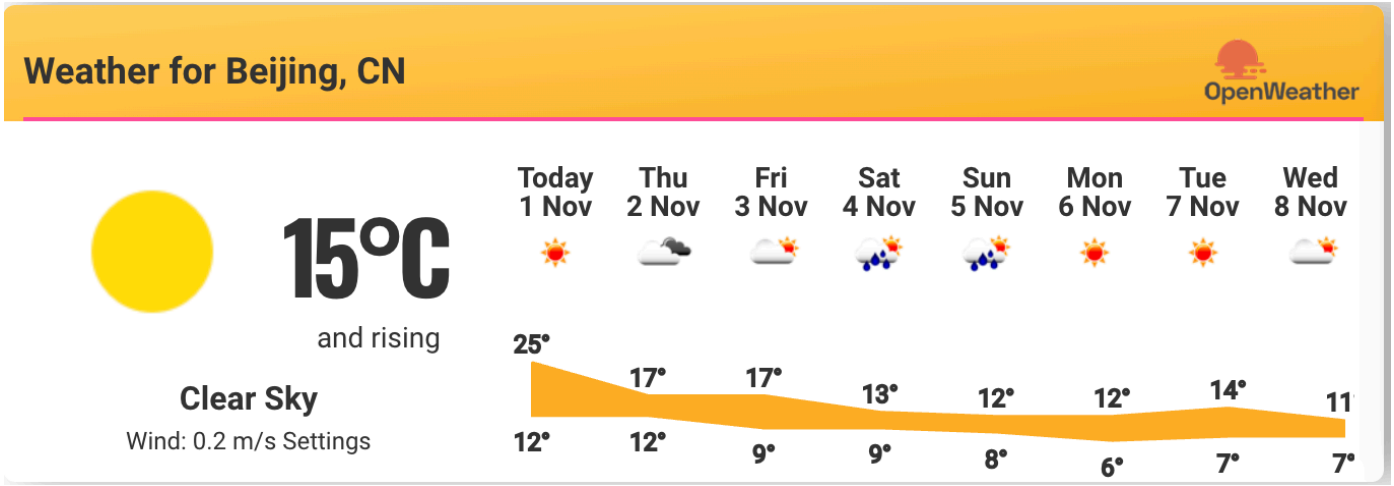
Java程序设计大作业

作业简介

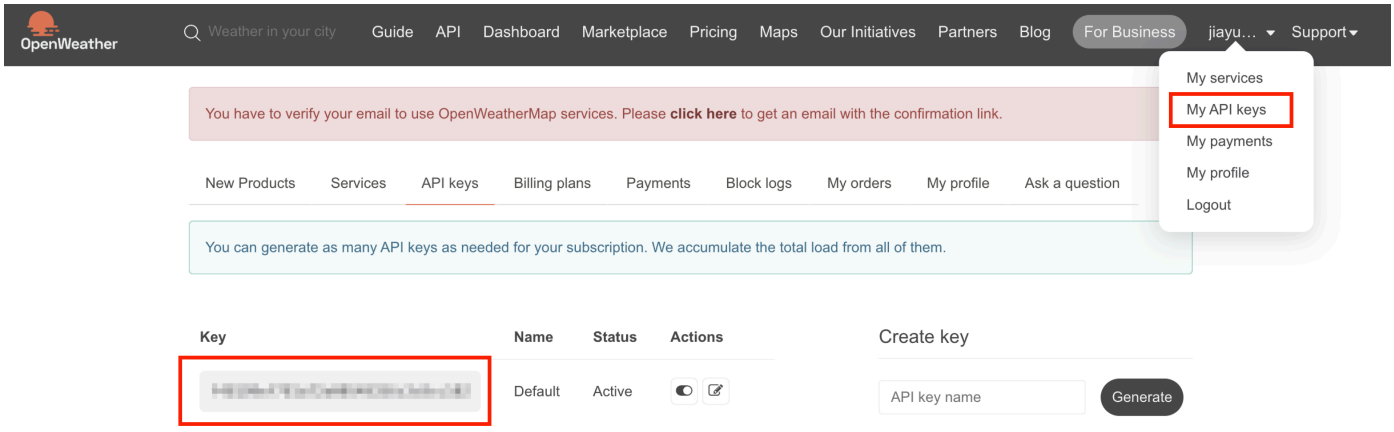
本次大作业的目标是实现一个天气预报应用程序`Weather`，利用天气预报数据的免费API：OpenWeatherMap，实现实时天气预报。下图是一个示例，不作为标准，具体的需求请参见`需求描述`。大作业满分100分，共分为3个STEP，建议同学按照顺序逐步实现。如果全部实现3个STEP，只需要提交STEP 3的程序；如果只实现了前几步，提交相应STEP的程序，获得前几步的相应分数。



注册API

本次作业要求利用天气预报数据API：OpenWeatherMap。

使用OpenWeatherMap API前需要先在官网注册。注册成功并登录后，需要在链接处获取访问API的key，用以在获取数据时认证身份，或者登录后在官网右上角的下拉菜单选择`My API keys`，生成并记录key。请注意API调用有频率限制，一分钟最多调用60次。



需求描述 - STEP 1：GUI (37'')

项目的第一步是实现天气预报应用的界面（GUI）。下面规定几种字号(size)，由大到小分别为H1>H2>H3>H4，具体的字号数值由同学自由设计。

1. 标题栏（5”）

- 在窗口最顶部实现一个标题，显示天气预报的城市和国家，在STEP 1中固定文字内容为`Weather for Beijing, CN`。（1″）
- 标题的字号为H2。（1″）
- 标题加粗。（1″）
- 标题栏有背景颜色，与窗口主体使用不同的背景，例如标题栏橙色，窗口其余部分白色；也可以使用图片背景，但需要与窗口主题作区分。（1″）
- 标题栏底部通过一个明显的边界线与其余部分区分，例如一个加粗的边框，但是标题栏左右和上侧没有边界线。（1″）

2. 当地时间（3″）

- 在标题栏中显示当地时间，位置在标题栏内部自由选择，时间精确到分钟(min)。（1″）
- 在STEP 1中规定文字内容为北京的当地时间（随时间变化）。（1″）
- 字号为H3，加粗。（1″）

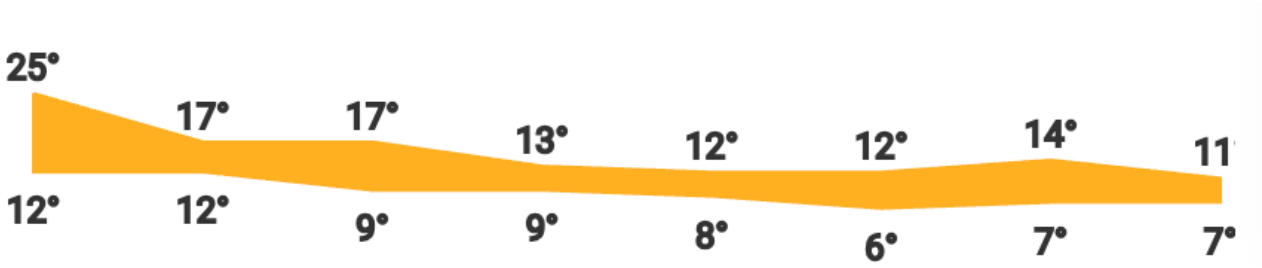
3. 当前天气（10″）

- 在窗口的某一部分划出一个**有边界**的矩形区域，显示当前天气。（1″）
- 矩形边框的边界设计为软矩形，亦即每个直角都是圆角。（1″）
- 矩形内任意位置显示温度，STEP 1文字内容为15°C，要打出摄氏度符号，H1，加粗。（2″）
- 矩形内任意位置显示天气状况，STEP 1文字内容为`Clear Sky`，首字母大写，H2，加粗。（1″）
- 矩形内任意位置显示风速，STEP 1文字内容为`Wind: 0.2 m/s`，H4，不加粗。（1″）
- 矩形内任意位置显示湿度，STEP 1文字内容为`Humidity: 56%`，H4，不加粗。（1″）
- 矩形内任意位置显示云量，STEP 1文字内容为`Clouds: 100%`，H4，不加粗。（1″）
- 矩形内任意位置显示天气状况图标，STEP 1图片内容为一个固定的Icon（点击链接）。（1″）
- 天气状况图标没有背景，需要与窗口背景融合，例如下图的情况是不允许的：（1″）

4. 未来天气（14″）

- 在窗口的某一部分划出一个**有边界**的矩形区域，显示未来天气，设计为软矩形边框。（1″）
- 矩形内有5列数据，分别对应今天和未来4天，这些数据需要对齐，包括星期、日期和天气状况图标。（1″）
- 每一列最上方显示星期。其中第一天显示`Tonight`，其余四天显示具体的星期，STEP 1 中需要根据北京当地时间来判断。星期的代码包括`Mon`,`Tue`,`Wed`,`Thu`,`Fri`,`Sat`,`Sun`。字号为H3，加粗。（2″）
- 每一列第二行显示日期，STEP 1中需要根据北京当地时间判断。日期的格式任意，但需要显示月和日，自由选择是否显示年。字号为H3，加粗。（2″）
- 每一列第三行显示天气状况图标，STEP 1中图片内容为一个固定的Icon（点击链接）。但是此处图标的大小必须小于`当前天气`中的图标。天气状况图标仍然没有背景。（2″）
- 每一列第四行显示每天的最大温度和最小温度。（6″）
    - 5天的温度共享一个坐标轴（不显示坐标轴），最下方是5天的最低温度，最上方是5天的最高温度。

（1″）
    - 将每天的最高温度和最低温度分别作出一个曲线图。（1″）
    - 最高温度和最低温度的曲线中间用任意颜色填充。（1″）
    - 在最高温度和最低温度曲线的上方和下方分别标出摄氏温度数值，自由选择是否标注单位。字号为H4，加粗。（1″）
    - 最高温度和最低温度的文字标注不能和曲线重合。（1″）
    - 每一列中的最高温度和最低温度的文字标注要对齐。（1″）



5. 空气污染（3″）

- 在窗口的任意位置标出空气污染水平，包括AQI指数，污染程度和主要污染物。STEP 1中固定文字为`AQI: 175`，`Moderatory Polluted`和`Primary Pollutants: PM2.5`。（2″）字号为H3，加粗。（1″）

请注意，所有的文字、图片和边框请不要重叠。（2″）

网络编程简介

超文本传输协议-HTTP

在STEP 1中，所有的天气数据都是静态的；而从STEP 2开始，我们将要通过网络API获取实时数据。这需要利用网络编程的数据传输协议：**HTTP（超文本传输协议）**。

HTTP是用于传输网页数据的协议。它建立在客户端-服务器模型上，简单来说，用户的计算机（客户端）通过浏览器发送请求给网站的服务器，然后服务器返回网页数据。HTTP请求有几种类型，我们要利用的是最简单的HTTP GET，主要用于获取服务器的资源，但是不向服务器传递大量数据的情景，例如我们的需求是从服务器请求实时天气数据。

现在我们就利用HTTP GET和浏览器完成一次OpenWeatherMap API的调用。请同学们在浏览器中输入下列的URL，注意URL最后的`{API KEY}`请替换为同学注册的key。

```
https://api.openweathermap.org/data/2.5/weather?
lat=39.906217&lon=116.3912757&units=metric&appid={API KEY}
```

浏览器会返回一个网页，其中包括的正是北京的实时天气数据：

```
{
    "base": "stations",
    "clouds": {
        "all": 100
    },
    "cod": 200,
    "coord": {
        "lat": 39.9062,
```

```
        "lon": 116.3913
    },
    "dt": 1699111133,
    "id": 1816670,
    "main": {
        "feels_like": 9.94,
        "grnd_level": 1015,
        "humidity": 56,
        "pressure": 1021,
        "sea_level": 1021,
        "temp": 9.94,
        "temp_max": 9.94,
        "temp_min": 9.94
    },
    "name": "Beijing",
    "sys": {
        "country": "CN",
        "id": 9609,
        "sunrise": 1699051548,
        "sunset": 1699089007,
        "type": 1
    },
    "timezone": 28800,
    "visibility": 10000,
    "weather": [
        {
            "description": "overcast clouds",
            "icon": "04n",
            "id": 804,
            "main": "Clouds"
        }
    ],
    "wind": {
        "deg": 78,
        "gust": 1.97,
        "speed": 1.19
    }
}
```


下面我们简单介绍这次API调用是如何发生的。首先，我们在浏览器中输入的URL由几部分组成，前半部分就是我们通常意义上的“网址”，它指定了我们使用的协议是HTTPS，并明确了天气数据资源所在的位置。

```
https://api.openweathermap.org/data/2.5/weather
```

URL的后半部分由`?`开始，它指定了本次请求的一些参数，包括天气预报的纬度(lat)、经度(lon)、温度的单位(units)、API Key(appid)。例如`lat=39.906217`就指定了纬度的值，而不同的参数通过`&`分隔开。这就组成了一个完整的URL。

```
?lat=39.906217&lon=116.3912757&units=metric&appid={API KEY}
```

浏览器得到URL后，会产生一次HTTP GET请求，送达OpenWeatherMap的服务器，服务器验证API Key通过后，就返回了我们需要的数据，服务器返回的内容称作HTTP Response。浏览器得到HTTP Response后，就将内容显示在网页上。

我们看到的网页内容是一段文本，不仅如此，我们发现数据内容具有一定的格式，这种格式称为JSON (JavaScript Object Notation)。下面我们将具体介绍JSON。在此之前，希望更多了解HTTP的同学请参考HTTP Tutorial。

JSON

JSON（JavaScript Object Notation）是一种轻量级的数据交换格式，易于人阅读和编写，同时也易于机器解析和生成。它是基于JavaScript语言标准的一个子集，但是它是完全独立于语言的，这意味着不仅仅是JavaScript，许多其他编程语言（例如Java）也都有解析和生成JSON数据的能力。JSON格式通常用于网络应用中，从服务器传送数据到客户端。

JSON的结构非常直观，它基于两种基本的数据结构：

1. 键值对(key-value)的集合：

- 由一些键（名称）和值组成，类似于Java中的Map数据结构。
- 在JSON中，这种结构表示为一个对象，用花括号`{}`包围。
- 键是字符串，值可以是字符串、数字、布尔值、数组、另一个JSON对象，或者`null`。

我们在天气数据的JSON数据中截取了一段，这就是一个简单的键值对集合。

```
{
    "base": "stations",
    "cod": 200,
    "dt": 1699111133,
    "id": 1816670
}
```

2. 有序列表：

- 类似于Java中的数组或List。
- 在JSON中，这种结构表示为一个数组，用方括号`[]`包围。
- 有序列表是值的有序集合，每个值又可以是任意类型（字符串、数字、其他有序列表或者键值对集合等）。

我们在天气数据的JSON数据中截取了一段，最外层是一个键值对集合，内部只有一个键值对，其中键是"weather"，值就是一个有序列表。有序列表中仅有一个元素，是一个键值对集合，里面刻画了天气的具体信息。

```
{
  "weather": [
        {
            "description": "overcast clouds",
            "icon": "04n",
            "id": 804,
            "main": "Clouds"
        }
    ]
}
```

可以发现，JSON数据中可以将多个键值对集合和有序列表列表嵌套组成，这使得我们可以便捷地从JSON数据中提取出我们需要的字段。例如在北京天气的JSON数据中，`weather.description`就是我们需要的天气状况描述。更多关于JSON的介绍请参见链接。

同学得到JSON格式的数据后，可以将其视作文本进行解析，利用课堂上文件读写所学习的方法，得到需要的字段。同学也可以引入成熟的JSON解析模块`org.json`，该模块的文档参见链接，并在链接处下载模块的打包文件`json-20231013.jar`。将模块导入为项目的依赖(dependency)后，就可以像使用JAVA标准库一样使用JSON解析模块。例如下面的示例代码：

```
import org.json.*;
public class Main {
    public static void main(String[] args) {
        JSONObject js = new JSONObject(
                "{\"city\":\"chicago\",\"name\":\"jon doe\",\"age\":\"22\"}"
        );
        System.out.println(js);
        String age = (String) js.opt("age");
        System.out.println(age);
    }
}
```

更多关于`org.json`的介绍参见链接1，链接2。

Java中的API调用实践

下面我们给出一次调用OpenWeatherMap API的实践。我们需要使用JAVA标准库中的`HttpURLConnection`类，文档参见链接。

1. 定义URL

```
String url = "https://api.openweathermap.org/data/2.5/weather?
lat=39.906217&lon=116.3912757&units=metric&appid={API KEY}";
URL obj = new URI(url).toURL();
```

2. 发出HTTP Request

```
HttpURLConnection con = (HttpURLConnection) obj.openConnection();
// Set request method to "GET"
con.setRequestMethod("GET");
```

3. 获取HTTP Response

```
int responseCode = con.getResponseCode();
if (responseCode != 200); // Process network exception.
// Read the response
BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
...
```

此处responseCode是HTTP协议中，服务器回传给客户端的一个状态码，用来指示客户端请求的处理结果。`responseCode==200`表示成功状态。如果`responseCode != 200`，表示网络状态存在异常，需要同学进行处理，例如尝试重新发送一次HTTP Request，或者弹出网络出错的提示。

需求描述 - STEP 2：网络编程(40'')

在STEP 2中，我们需要调用OpenWeatherMap API获取实时的天气数据。但是在STEP 2，我们暂时只预测北京当地的天气。首先我们规定需要使用的API接口。

API接口

1. 实时天气

实时天气API的文档请参见链接。下面是API调用的示例URL：

```
https://api.openweathermap.org/data/2.5/weather?
lat=39.906217&lon=116.3912757&units=metric&appid={API KEY}
```

各个参数为:

- lat：纬度，北京纬度为39.906217
- lon: 经度，北京经度为116.3912757
- units: 单位，metric表示摄氏度
- appid: API Key

2. 未来5天天气预测

给出**未来**5天，每3小时一次的天气预测，总共40个时间点。API的文档请参见链接。下面是API调用的示例URL：

```
http://api.openweathermap.org/data/2.5/forecast?
lat=39.906217&lon=116.3912757&units=metric&appid={API KEY}
```

各个参数为:

- lat：纬度，北京纬度为39.906217
- lon: 经度，北京经度为116.3912757
- units: 单位，metric表示摄氏度

- appid: API Key

3. 空气污染

给出当前的空气污染监测结果。API的文档请参见链接。下面是API调用的示例URL：

```
http://api.openweathermap.org/data/2.5/air_pollution?
lat=39.906217&lon=116.3912757&appid={API KEY}
```

各个参数为:

- lat：纬度，北京纬度为39.906217
- lon: 经度，北京经度为116.3912757
- appid: API Key

STEP 2

首先我们给出功能需求。

1. 当前天气（13”）

- 矩形内任意位置显示温度，STEP 2文字内容为实时天气API：`main.temp`字段。要求使用摄氏度单位。（2”）
- 矩形内任意位置显示天气状况，STEP 2文字内容为实时天气API：`weather.description`字段，注意`weather`字段是一个有序列表，可能包含多种天气状况，其中列表的第一个元素是主要天气，请显示主要天气。（2”）
- 矩形内任意位置显示风速，STEP 2文字内容为实时天气API：`wind.speed`字段，单位为`m/s`。（2”）
- 矩形内任意位置显示湿度，STEP 2文字内容为实时天气API：`main.humidity`字段。（2”）
- 矩形内任意位置显示云量，STEP 2文字内容为实时天气API：`clouds.all`字段。（2”）
- 矩形内任意位置显示天气状况图标，STEP 2图片内容需要根据实时天气状况进行判断。利用实时天气API：`weather.icon`字段，获取实时天气状态的图标ID，然后通过链接获取图标ID对应的图标PNG的URL。（3”）例如，`weather.icon=='04n'`，可以查询得到对应图标的URL为https://openweathermap.org/img/wn/04n@2x.png。下载得到对应的ICON：

2. 未来天气（10”）

- 天气状况图标，STEP 2中图片内容需要根据实时天气状况进行判断。利用未来5天天气预测API: `list.weather.icon`字段，选择对应的图标。请注意，`list`是一个有序列表，包含了40个时间点的天气预测。
  - 对于Tonight来说，如果在`list`中包含**当地时间今天**18:00的数据，就给出该时间点的预测。否则显示`Data Unavailbale`，字体任意，并且也不显示今晚的最高温度和最低温度数据，以及曲线图。（3”）
  - 对于未来4天来说，选择`list`中该日**当地时间**中午12:00的数据。（2”）
- 最大温度和最小温度，STEP 2中利用未来5天天气预测API: `list.main.temp`字段，给出最高温度和最低温度的预测。

- 对于Tonight来说，如果在`list`中包含**当地时间今天**18:00的数据，就统计`list`中时间范围在**当地时间今天**`24:00`前的数据（不包括24:00），得到的最小值和最大值分别作为最小温度和最大温度的预测。（3”）否则不显示今晚的最高温度和最低温度数据，以及曲线图。
- 对于未来4天来说，统计`list`中时间范围在该日**当地时间**`0:00-24:00`间的数据（包括00:00，不包括24:00），得到的最小值和最大值分别作为最小温度和最大温度的预测。（2”）

3. 空气污染（10”）

下面首先给出中国空气质量AQI指数的计算方法，详情参考链接：

首先给出AQI指数和污染程度(Air pollution category)的对应表格。

AQI	Air pollution level	Air pollution category
0-50	Level 1	Excellent
51-100	Level 2	Good
101-150	Level 3	Lightly Polluted
151-200	Level 4	Moderatory Polluted
201-300	Level 5	Heavily Polluted
>300	Level 6	Severely Polluted

那么AQI指数是如何计算的呢？AQI是由每个污染物的个体指数（Individual Air Quality Index, IAQI）综合得到的。具体来说，AQI就是所有污染物的IAQI的最大值，而IAQI值最大的污染物则为主要污染物，若IAQI值相等且最大的污染物有多个，那么它们都是主要污染物。

IAQI是使用以下公式和断点浓度表计算的：

```
I = (I_high - I_low) / (C_high - C_low) * (C - C_low) + I_low
```

其中：

- `I` 是个体空气质量指数，
- `C` 是污染物浓度，
- `C_low` 是浓度断点≤C，
- `C_high` 是浓度断点＞C，
- `I_low` 是对应于 `C_low` 的指数断点，
- `I_high` 是对应于 `C_high` 的指数断点。

下面是断点浓度表：

IAQI	SO_2	NO_2	PM_{10}	CO	O_3	$PM_{2.5}$
0	0	0	0	0	0	0
50	150	100	50	5	160	35
100	500	200	150	10	200	75
150	650	700	250	35	300	115
200	800	1200	350	60	400	150
300	1600	2340	420	90	800	250
400	2100	3090	500	120	1000	350
500	2620	3840	600	150	1200	500

污染物的浓度单位均是μg/m³，除了CO，单位是mg/m³。**请注意比对API返回数据中的单位。**

由于完全准确的IAQI计算方法需要使用24小时监测数据，在OpenWeatherMap API中缺失，因此上述计算得到的AQI与官方数据可能存在偏差。

下面是空气污染水平的功能需求：

使用空气污染API：`list.components`字段得到的各污染物浓度。

- 计算AQI指数。（6"）
- 汇报污染程度。（2"）
- 汇报主要污染物。（2"）

接下来我们给出STEP 2的性能需求：（7"）

1. 出现网络错误时程序不崩溃，并弹出相应提示，网络恢复后能够自动获取和更新数据。（3"）
2. 页面上的数据可以自动更新，例如当前时间、天气数据的更新。（2"）
3. 界面不卡顿，图标和数据加载没有延迟。（2"）

需求描述 - STEP 3：切换城市(19")

在STEP 3中，我们要求可以给出全球城市的天气预测。这需要使用新的API接口。

API接口

1. 地理位置查询API：

地理位置查询API的文档请参见链接。下面是API调用的示例URL：

```
https://api.openweathermap.org/geo/1.0/direct?q=Beijing&limit=5&appid={API KEY}
```

各个参数为:

- q：城市名称
- limit: 搜索返回的待选城市列表

- appid: API Key

STEP 3

在STEP 3中，我们允许用户切换城市并取得当地的天气预测。

首先介绍功能需求：

1. 城市搜索（5”）

- 在界面任意位置自由设计一个文本框，输入城市名称（英文），然后弹出一个下拉列表。（2”）
- 下拉列表内容为地理位置查询API的返回结果中的**前3个**结果，下拉列表每一行显示的字段为`name,state,country,lat,lon`。其中state字段若不存在则省略。lat和lon字段根据正负值判断为北纬、南纬和东经、西经，分别用N,S,E,W表示。（2”）例如：

```
Beijing, Beijing, CN, 39.9N, 116.4E
```

- 搜索不到结果则不显示下列表，结果不满3个就显示全部搜索到的内容。（1”）

2. 城市天气（10”）

- 在下拉列表中点击对应的选项，跳转到对应城市的天气预报。标题栏中的城市、国家需要进行切换。（2”）
- 当地时间需要进行切换，根据实时天气API的`timezone`字段判断时区并计算当地时间。注意不要直接根据经纬度计算，可能不准确。（2”）
- 更新当前天气。（2”）
- 更新未来5天的天气预测。（2”）
- 更新空气污染数据。（2”）

下面介绍性能需求：（4”）

1. 城市搜索的过程流畅不卡顿，切换城市天气界面的过程流畅不卡顿。（2”）
2. 若城市天气的某些字段在API返回数据中不存在，程序不崩溃，应当在数据的位置填充`Unavailable`等进行替代。（2”）

实验报告(4'')

实验报告为必选，包括以下内容：

1. 描述程序操作方式(1'')：说明实现到的STEP和使用的SDK版本。（可选）对于复杂和较难操作的功能，说明使用应用程序的方法，目的是使程序测试者可以发现并测试实现的每一个功能。

2. 使用类图或者自然语言描述项目的架构与设计(3”)。
3. （可选）其他任何希望助教了解的内容，例如说明作业完成过程中参考的资料或个人。

补充说明

1. 对于没有明确描述的功能或实现效果，选择任意方式实现均可。
2. 提交所有源代码，实验报告和一个可以直接运行的 jar 包（位置在根目录，缺少 jar 包大作业总分扣除10分），并打包为一个zip文件。其中jar包中需要包含所有的依赖（引入的外部模块）和资源（图片），这样的jar又称fat jar。

```
└2000123456_张三.zip
      ├ report.pdf //实验报告
      ├ ***.jar    //可执行文件
      └ src        //源代码目录
```

3. 使用JAVA语言编写程序，不限制使用的模块和库。
4. 编程作业为个人作业，请独立完成。如果提交的代码参考了资料或个人，请在report中标注来源，否则视为抄袭，记0分。