第5章 Chapter 5

小程序网络API·天气查询

本章主要介绍使用小程序网络 API 的相关应用制作一款天气查询小程序。

本章学习目标

- 掌握服务器域名配置和临时服务器部署；
- 掌握 wx.request 接口的用法。

5.1 准备工作

视频讲解

5.1.1 API 密钥申请

本小节主要介绍如何申请获得开源 API 的密钥。这里选择了可以提供全球气象数据服务接口的和风天气 API，其官方网址为“https://www.heweather.com/”（如图 5-1 所示）。

图 5-1 “和风天气”官方主页（访问时间：2018.10.27 10:26）

用户选择“免费用户”类型，使用邮箱进行注册并激活后可以获取三天之内全球各地区的实时天气，免费接口调用流量为 1000 次/天、频率为 200 次/分钟，该数据基本上可以满足读者的开发学习需求。

注册完毕之后可以访问“https://console.heweather.com/my/service”来查看账号信息，用户登录后即可看到开发者申请到的个人认证 key，如图 5-2 所示。

开发者需记录上述页面中的个人认证 key，该信息在小程序发出网络请求时会作为身份识别的标识一并发送给和风天气的第三方服务器。至此，开源 API 的密钥申请就已经顺利完成，读者可以进行 5.1.2 节的学习，了解如何调用 API 获取气象数据。

图 5-2 个人认证 key 查询页面（访问时间：2018.10.27 10:27）

5.1.2 API 调用方法

视频讲解

目前免费用户可以调用的最新版接口地址为“https://free-api.heweather.com/s6/”，其服务器节点在中国境内。该接口地址后面追加不同的关键词将获取不同种类的气象数据信息，例如 alarm 为天气自然灾害预警。读者可以访问官方文档（https://dev.heweather.com/docs/api/）了解各类关键词的使用方法。

本示例将选用关键词 weather 进行实况天气数据的获取。实况天气即为当前时间点的天气状况以及温/湿/风/压等气象指数，具体包含体感温度、实测温度、天气状况、风力、风速、风向、相对湿度、大气压强、降水量、能见度等。目前该接口允许查询的城市覆盖范围为全球任意一个城市。

基于关键词 weather 的接口具有两个必填参数和两个可选参数，如表 5-1 所示。

表 5-1 weather 接口参数一览表

参数名称	参数类型	解释
city	必填参数	用于规定需要查询的城市，可以填入城市名称（国内城市填中文或拼音均可）、城市 ID、IP 地址或经纬度。 例如： city=北京、city=beijing（城市名称） city=CN101010100（城市 ID） city=60.194.130.1（IP 地址） city=120.343,36.088（经纬度）
key	必填参数	需要填入用户的个人认证 key 字符串。接口将通过该数据判断是否为授权用户，并可以进一步判断是否为付费用户。 例如：key=123abc456dfg
lang	可选参数	用于指定数据的语言版本，不添加 lang 参数则默认为简体中文。 例如：lang=en 需要注意的是，国内某些特定数据（例如生活指数、空气质量等）不支持多语言版
unit	可选参数	单位选择，公制（m）或英制（i），默认为公制单位。 例如：unit=i 详见表 5-2“度量衡单位一览表”

其中与 unit 参数相关的公制和英制单位对比如表 5-2 所示。

表 5-2 度量衡单位一览表

数 据 项	公 制 单 位	英 制 单 位
温度	摄氏度：℃	华氏度：℉
风速	公里/小时：km/h	英里/小时：mile/h
能见度	公里：km	英里：mile
大气压强	百帕：hPa	百帕：hPa
降水量	毫米：mm	毫米：mm
PM2.5	微克/立方米：$\mu g/m^3$	微克/立方米：$\mu g/m^3$
PM10	微克/立方米：$\mu g/m^3$	微克/立方米：$\mu g/m^3$
O_3	微克/立方米：$\mu g/m^3$	微克/立方米：$\mu g/m^3$
SO_2	微克/立方米：$\mu g/m^3$	微克/立方米：$\mu g/m^3$
CO	毫克/立方米：mg/m^3	毫克/立方米：mg/m^3
NO_2	微克/立方米：$\mu g/m^3$	微克/立方米：$\mu g/m^3$

注意：部分数据项无论选择何种单位均会使用公制单位。

免费用户调用接口的常见语法格式如下：

```
https://free-api.heweather.com/s6/weather/now?[parameters]
```

其中[parameters]需要替换成使用到的参数，多个参数之间使用 & 符号隔开。

例如，使用拼音查询上海市天气数据的写法如下：

```
https://free-api.heweather.com/s6/weather/now?location=shanghai&key=1234abcd
```

注意，其中 key 的值 1234abcd 为随机填写的内容，请在实际开发中将其替换为真实的个人认证 key，否则接口将无法获取数据。

用户可以直接将这段地址输入到浏览器的地址栏中测试数据返回结果，如图 5-3 所示。

{"HeWeather6":[{"basic":{"cid":"CN101020100","location":"上海","parent_city":"上海","admin_area":"上海","cnty":"中国","lat":"31.23170662","lon":"121.47264099","tz":"+8.00"},"update":{"loc":"2018-10-27 10:45","utc":"2018-10-27 02:45"},"status":"ok","now":{"cloud":"0","cond_code":"100","cond_txt":"晴","fl":"17","hum":"19","pcpn":"0.0","pres":"1024","tmp":"19","vis":"10","wind_deg":"315","wind_dir":"西北风","wind_sc":"1","wind_spd":"4"}}]}

图 5-3 免费天气查询接口返回结果页面（访问时间：2018.10.27 10:59）

由该图可知，指定城市的天气数据返回结果是 JSON 数据格式的文本内容，其中包含的数据是以“名称:值”的形式存放。

为方便用户查看，将图 5-3 返回的数据内容整理格式如下：

```
{
  "HeWeather6":[
    {
      "basic":{
        "cid":"CN101020100",
        "location":"上海",
        "parent_city":"上海",
        "admin_area":"上海",
        "cnty":"中国",
```

```
        "lat":"31.23170662",
        "lon":"121.47264099",
        "tz":"+8.00"
      },
      "update":{
        "loc":"2018-10-27 10:45",
        "utc":"2018-10-27 02:45"
      },
      "status":"ok",
      "now":{
        "cloud":"0",
        "cond_code":"100",
        "cond_txt":"晴",
        "fl":"17",
        "hum":"19",
        "pcpn":"0.0",
        "pres":"1024",
        "tmp":"19",
        "vis":"10",
        "wind_deg":"315",
        "wind_dir":"西北风",
        "wind_sc":"1",
        "wind_spd":"4"
      }
    }
  ]
}
```


返回的字段说明如表 5-3 所示。

表 5-3 实况天气返回字段说明

basic：基础信息

参　数	描　述	示 例 值
location	地区/城市名称	海淀
cid	地区/城市 ID	CN101080402
lat	地区/城市纬度	39.956074
lon	地区/城市经度	116.310316
parent_city	该地区/城市的上级城市	北京
admin_area	该地区/城市所属行政区域	北京
cnty	该地区/城市所属国家名称	中国
tz	该地区/城市所在时区	8

update：接口更新时间

参　数	描　述	示 例 值
loc	当地时间，24 小时制，格式为 yyyy-MM-dd HH:mm	2017/10/25 12:34
utc	UTC 时间，24 小时制，格式为 yyyy-MM-dd HH:mm	2017/10/25 4:34

now：实况天气

参　数	描　述	示 例 值
fl	体感温度，默认单位为摄氏度	23
tmp	温度，默认单位为摄氏度	21
cond_code	实况天气状况代码	100
cond_txt	实况天气状况描述	晴

续表

参　数	描　述	示 例 值
wind_deg	风向360角度	305
wind_dir	风向	西北
wind_sc	风力	3
wind_spd	风速,公里/小时	15
hum	相对湿度	40
pcpn	降水量	0
pres	大气压强	1020
vis	能见度,默认单位为公里	10
cloud	云量	23

status：接口状态

参　数	描　述	示 例 值
status	接口状态,具体含义请参考表5-4“接口状态码及错误码说明”	ok

其中参数status的状态码及错误码说明如表5-4所示。

表5-4　接口状态码及错误码说明

代　码	说　明
ok	数据正常
invalid key	错误的key,请检查key是否输入以及是否输入有误
unknown location	未知或错误城市/地区
no data for this location	该城市/地区没有所请求的数据
no more requests	超过访问次数,需要等到当月最后一天24点(免费用户为当天24点)后进行访问次数的重置或升级访问量
param invalid	参数错误,请检查传递的参数是否正确
too fast	超过限定的QPM,请参考QPM说明
dead	无响应或超时,接口服务异常请联系产品开发商
permission denied	无访问权限,没有购买所访问的这部分服务
sign error	签名错误,请参考签名算法

如果接口无法正确地获取数据,可以根据状态码对比该表查询原因。

用户可以根据指定的名称找到对应的数据值,例如在实况天气数据(now)中可以查到当前城市的温度,对应的字段节选如下：

```
"tmp":"19"
```

上述代码表示当前城市的温度为19℃。

5.1.3　服务器域名配置

视频讲解

每一个小程序在与指定域名地址进行网络通信前都必须将该域名地址添加到管理员后台白名单中,因此本示例需要对域名地址“https://free-api.heweather.com”进行服务器配置。

小程序开发者登录mp.weixin.qq.com进入管理员后台,单击“设置”按钮,切换至“开发设置”

面板，在“服务器域名”栏中可以添加或修改需要进行网络通信的服务器域名地址，如图 5-4 所示。

图 5-4　服务器域名配置

将当前需要使用的接口添加到“request 合法域名”中，配置完成后再登录小程序开发工具就允许小程序与指定的服务器域名地址之间的网络通信了，注意每个月只可以申请修改 5 次服务器域名配置。

5.2　项目创建

视频讲解

本项目创建选择空白文件夹 weatherDemo，效果如图 5-5 所示。

图 5-5　小程序项目填写效果示意图

单击“新建”按钮完成项目创建，然后准备手动创建页面配置文件。

5.3 页面配置

视频讲解

5.3.1 创建页面文件

项目创建完毕后，在根目录中会生成文件夹 pages 用于存放页面文件。一般来说首页默认命名为 index，表示小程序运行的第一个页面；其他页面名称可以自定义。本项目只需要保留首页(index)即可。

具体操作如下：

(1) 将 app.json 文件内 pages 属性中的“pages/logs/logs”删除，并删除上一行末尾的逗号。

(2) 按快捷键 Ctrl+S 保存当前修改。

5.3.2 删除和修改文件

具体操作如下：

(1) 删除 utils 文件夹及其内部所有内容。

(2) 删除 pages 文件夹下的 logs 目录及其内部所有内容。

(3) 删除 index.wxml 和 index.wxss 中的全部代码。

(4) 删除 index.js 中的全部代码，并且输入关键词 page 找到第二个选项按回车键让其自动补全函数(如图 5-6 所示)。

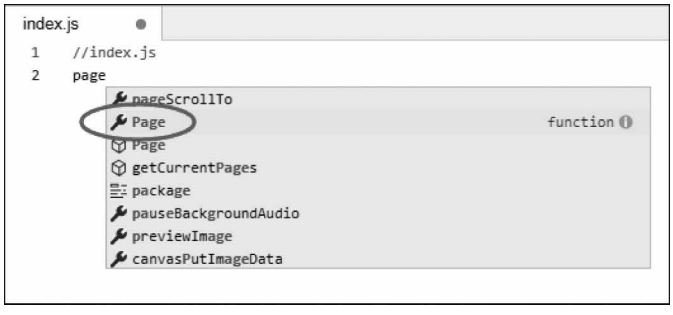

图 5-6 输入关键词创建 Page 函数

(5) 删除 app.wxss 中的全部代码。

(6) 删除 app.js 中的全部代码，并且输入关键词 app 找到第一个选项按回车键让其自动补全函数(如图 5-7 所示)。

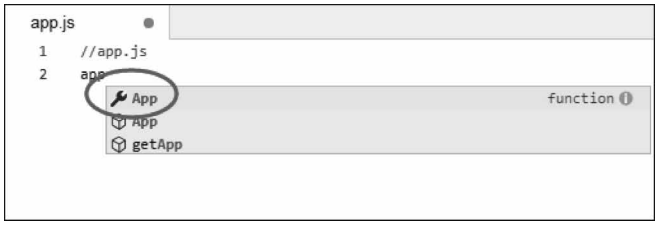

图 5-7 输入关键词创建 App 函数

5.3.3 创建其他文件

接下来创建其他自定义文件，本项目还需要一个文件夹用于存放天气图标素材。文件夹名称由开发者自定义（例如 images），单击目录结构左上角的＋号创建文件夹并命名为 images。

本项目用到的图标素材共计 75 个，均来源于和风天气官网，图标素材展示如图 5-8 所示。

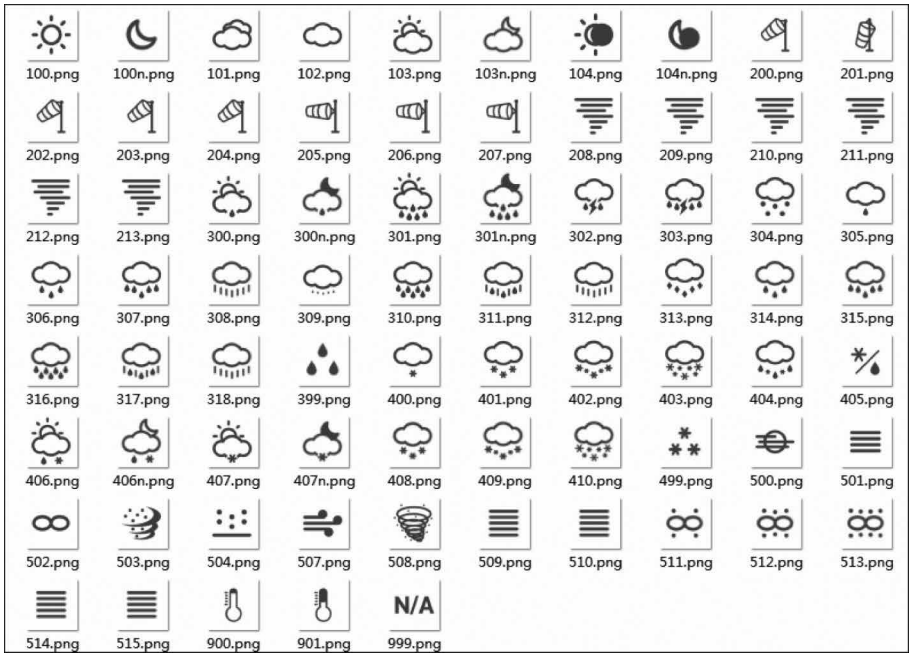

图 5-8 天气图标素材展示

其中图标文件名为对应的天气代码，扩展名均为. png。需要注意的是，部分图标文件名带有字母 n，表示夜间天气图标，例如 100n. png。

右击目录结构中的 images 文件夹，选择“硬盘打开”，在该文件夹中新建二级目录 weather_icon，然后将图标文件全部复制、粘贴进去。完成后的目录结构如图 5-9 所示。

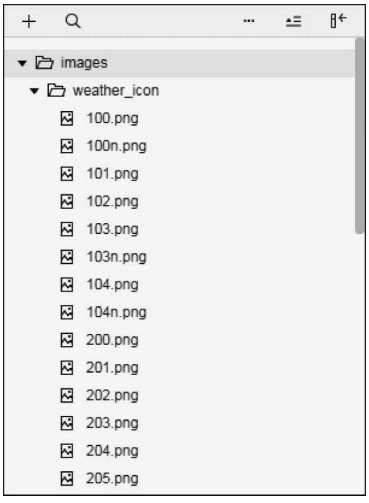

图 5-9 页面文件创建完成

此时文件配置就全部完成，5.4 节将正式进行页面布局和样式设计。

5.4 视图设计

视频讲解

5.4.1 导航栏设计

小程序默认导航栏是黑底白字的效果，因此需要在 app.json 中自定义导航栏标题和背景颜色。更改后的 app.json 文件代码如下：

```
1.  {
2.    "pages": [
3.      "pages/index/index"
4.    ],
5.    "window": {
6.      "navigationBarBackgroundColor": "#3883FA",
7.      "navigationBarTitleText": "今日天气"
8.    }
9.  }
```

上述代码可以更改所有页面的导航栏标题文本为“今日天气”、背景颜色为蓝色(#3883FA)。预览效果如图 5-10 所示。

图 5-10 自定义导航栏效果

5.4.2 页面设计

视频讲解

页面上主要包含 4 个区域，具体内容解释如下。

- 区域 1：地区选择器，用户可以自行选择查询的省、市、区；
- 区域 2：显示当前城市的温度和天气状态的文字说明；
- 区域 3：显示当前城市的天气图标；
- 区域 4：分多行显示其他天气信息，例如湿度、气压、能见度和风向等。

注意，面板之间需要有一定的间隔距离，设计图如图 5-11 所示。

计划使用如下组件。

- 页面整体：<view>组件，并定义 class='container'；
- 区域 1：<picker>组件；
- 区域 2：<text>组件；
- 区域 3：<image>组件；
- 区域 4：<view>组件，并定义 class='detail'；
- 区域 4 内单元行：4 个<view>组件，并定义 class='bar'；
- 区域 4 内单元格：每行 3 个<view>组件，并定义 class='box'。

1 整体容器设计

首先定义页面容器(<view>)，WXML(pages/index/index.wxml)代码片段如下：

```
1.  <view class='container'>
2.  </view>
```

在 app.wxss 中设置容器样式，代码片段如下：

```
/* 背景容器样式 */
.container{
  height: 100vh;                    /* 高度为 100 视窗,写成 100% 无效 */
  display: flex;                    /* flex 布局模型 */
  flex-direction: column;           /* 垂直布局 */
  align-items: center;              /* 水平方向居中 */
  justify-content:space-around;     /* 调整间距 */
}
```

当前效果如图 5-12 所示。

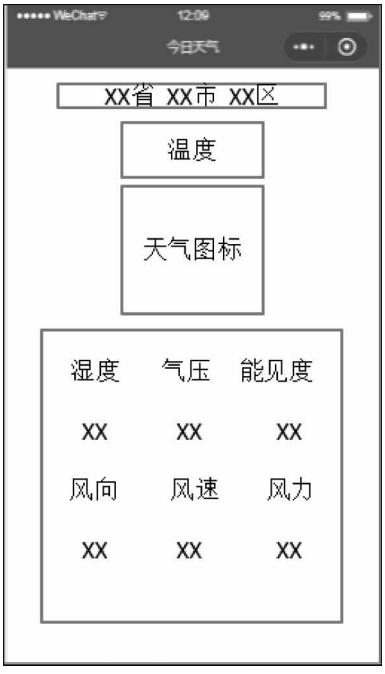

图 5-11 页面设计图

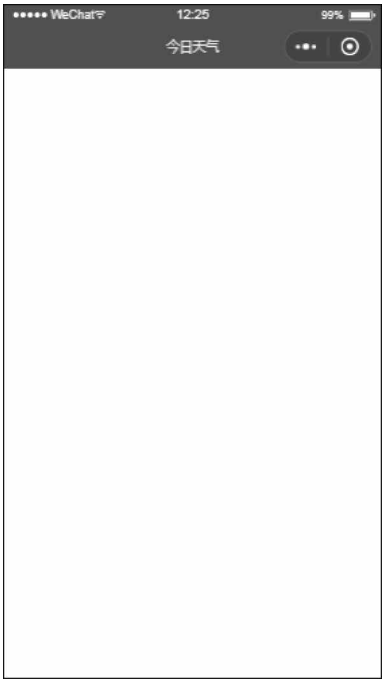

图 5-12 页面预览效果

由于还没有添加组件元素,所以尚看不出来 flex 布局模型效果。

2 区域 1(地区选择器)设计

区域 1 需要使用< picker >组件来实现一个地区选择器,用户点击可切换选择其他城市。WXML(pages/index/index.wxml)代码片段修改如下:

```
<view class = 'container'>
  <!-- 区域 1: 地区选择器 -->
  <picker mode = 'region'>
    <view>北京市</view>
  </picker>
</view>
```

< picker >组件内部是开发者任意填写的一个城市名称,当前效果如图 5-13 所示。

由图可见,点击城市名称时会从底部弹出控件,用户可以进行省、市、区的选择。

3 区域 2(文本)设计

区域 2 需要使用< text >组件实现一个单行天气信息,包括当前城市的温度和天气状况。WXML(pages/index/index.wxml)代码片段修改如下:

(a) 页面初始效果　　(b) 点击城市名称时的效果

图 5-13　区域 1 预览效果

```
1.  <view class = 'container'>
2.    <!-- 区域 1: 地区选择器 -->
3.    …
4.    <!-- 区域 2: 单行天气信息 -->
5.    <text>19℃ 晴</text>
6.  </view>
```

WXSS(pages/index/index.wxss)代码片段如下：

```
1.  /* 文本样式 */
2.  text{
3.    font-size: 80rpx;
4.    color: #3C5F81;
5.  }
```

当前效果如图 5-14 所示。

当前显示的文本内容由开发者自定义，待查询到实际数据后将动态更新文本内容。

4 区域 3(天气图标)设计

区域 3 需要使用<image>组件展示当前城市的天气图标。

WXML(pages/index/index.wxml)代码片段修改如下：

```
1.  <view class = 'container'>
2.    <!-- 区域 1: 地区选择器 -->
3.    …
4.    <!-- 区域 2: 单行天气信息 -->
5.    …
6.    <!-- 区域 3: 天气图标 -->
7.    <image src = '/images/weather_icon/999.png' mode = 'widthFix'></image>
8.  </view>
```

WXSS(pages/index/index.wxss)代码片段如下：

```
/* 图标样式 */
image{
  width: 220rpx;
}
```

当前效果如图 5-15 所示。

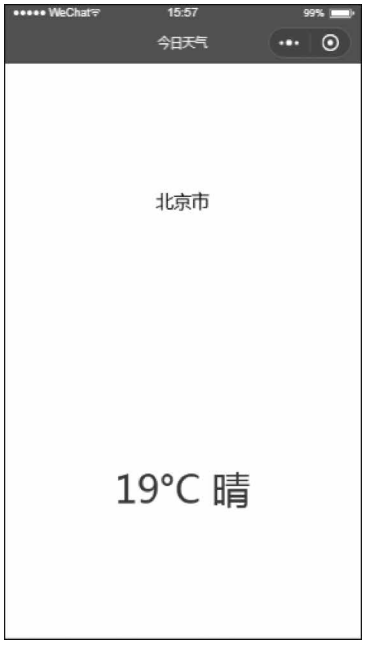

图 5-14　区域 2 预览效果

图 5-15　区域 3 预览效果

“N/A”表示天气状况为“未知”,待查询到实况数据后将动态更新图标内容。

5 区域 4(多行天气信息)设计

区域 4 需要使用< view >组件展示多行天气信息。

WXML(pages/index/index.wxml)代码片段修改如下：

```
<view class = 'container'>
  <!-- 区域 1: 地区选择器 -->
  …
  <!-- 区域 2: 单行天气信息 -->
  …
  <!-- 区域 3: 天气图标 -->
  …
  <!-- 区域 4: 多行天气信息 -->
  <view class = 'detail'>
    <view class = 'bar'>
      <view class = 'box'>湿度</view>
      <view class = 'box'>气压</view>
      <view class = 'box'>能见度</view>
    </view>
    <view class = 'bar'>
      <view class = 'box'> 0 % </view>
      <view class = 'box'> 0 hPa </view>
```

```
18.        <view class = 'box'> 0 km </view>
19.      </view>
20.      <view class = 'bar'>
21.        <view class = 'box'>风向</view>
22.        <view class = 'box'>风速</view>
23.        <view class = 'box'>风力</view>
24.      </view>
25.      <view class = 'bar'>
26.        <view class = 'box'> 0 </view>
27.        <view class = 'box'> 0 km/h </view>
28.        <view class = 'box'> 0 级</view>
29.      </view>
30.    </view>
31. </view>
```

WXSS(pages/index/index.wxss)代码片段如下：

```
1.  /* 区域 4 整体样式 */
2.  .detail{
3.    width: 100 %;
4.    display: flex;
5.    flex-direction: column;
6.  }
7.  /* 区域 4 单元行样式 */
8.  .bar{
9.    display: flex;
10.   flex-direction: row;
11.   margin: 20rpx 0;
12. }
13. /* 区域 4 单元格样式 */
14. .box{
15.   width: 33.3 %;
16.   text-align: center;
17. }
```

当前效果如图 5-16 所示。

当前为开发者自定义数据，待查询到实况数据后将动态更新区域 4 的内容。此时页面设计就全部完成了，接下来需要进行逻辑实现。

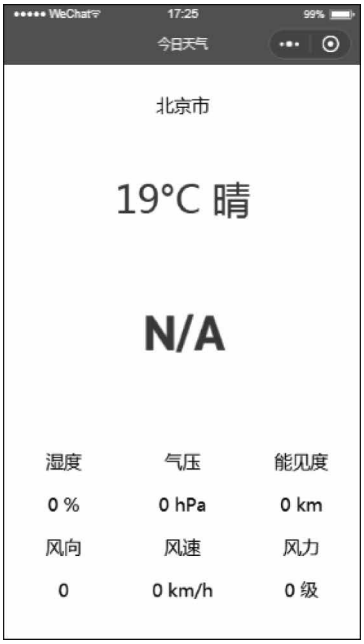

图 5-16 区域 4 预览效果

5.5 逻辑实现

视频讲解

5.5.1 更新省、市、区信息

首先修改< picker >组件中的“北京市”为{{region}}，然后为< picker >组件追加自定义 bindchange 事件，用于监听选项变化。

WXML(pages/index/index.wxml)代码片段修改如下：

```
1.  <view class = 'container'>
2.    <!-- 区域 1: 地区选择器 -->
3.    <picker mode = 'region' bindchange = 'regionChange'>
4.      <view>{{region}}</view>
5.    </picker>
6.  </view>
```

由于地区选择器的返回结果是数组的形式，所以在 JS 文件的 data 中定义 region 为包含了省、市、区 3 个项目的数组，初始城市信息由开发者自定义。

JS(pages/index/index.js)代码片段修改如下：

```
1.  Page({
2.    /**
3.      * 页面的初始数据
4.      */
5.    data: {
6.      region:['安徽省','芜湖市','镜湖区']
7.    },
8.    /**
9.      * 更新省、市、区信息
10.     */
11.   regionChange: function(e) {
12.     this.setData({region: e.detail.value});
13.   },
14. })
```

运行效果如图 5-17 所示。

(a) 重新选择城市　　(b) 更新省、市、区信息

图 5-17　更新省、市、区信息前后

由图可见，当前已经可以自行切换到国内任意省、市、区。

5.5.2　获取实况天气数据

视频讲解

在 JS 文件中使用自定义函数 getWeather 进行实况天气数据的获取。由于非直辖市无法查询到具体的区，所以后续的天气查询以城市作为查询依据。

JS(pages/index/index.js)代码片段修改如下：

```
Page({
  /**
   * 获取实况天气数据
   */
  getWeather: function() {
    var that = this;
    wx.request({
      url: 'https://free-api.heweather.com/s6/weather/now',
      data:{
        location:that.data.region[1],
        key:'请填入开发者申请的和风天气密钥'     //替换成开发者申请到的 key
      },
      success:function(res){
        console.log(res.data);
      }
    })
  },
})
```

将上述函数在生命周期函数 onLoad 和自定义函数 regionChange 中分别进行调用，表示当页面加载时和切换城市时均主动获取一次实况天气数据。

JS(pages/index/index.js)代码片段修改如下：

```
Page({
  /**
   * 更新省、市、区信息
   */
  regionChange: function(e) {
    this.setData({region: e.detail.value});
    this.getWeather();                    //更新天气
  },
  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function(options) {
    this.getWeather();                    //更新天气
  },
})
```

在联网状态下保存后重新运行会在 Console 控制台得到第三方服务器发回的 JSON 数据，如图 5-18 所示。

图 5-18 Console 控制台获取到服务器返回数据

由图可见，实况天气数据包含在 HeWeather6[0].now 属性中。更新 getWeather 函数，将该属性存到 JS 文件的 data 中，JS(pages/index/index.js)代码片段修改如下：

```
1.  Page({
2.    /**
3.     * 获取实况天气数据
4.     */
5.    getWeather: function() {
6.      var that = this;
7.      wx.request({
8.        url: 'https://free-api.heweather.com/s6/
  weather/now',
9.        data:{
10.         location:that.data.region[1],
11.         key:'f0671b6589ff43019e72970d334ea93e'
12.       },
13.       success:function(res){
14.         that.setData({now:res.data.HeWeather6[0].now});
15.       }
16.     })
17.   },
18. })
```

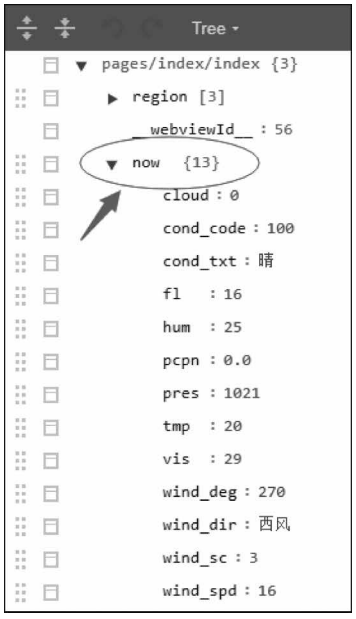

图 5-19 AppData 面板获取到数据

此时重新运行将在 AppData 面板中查到已经被记录的天气数据，如图 5-19 所示。

之后只需要将这些数据更新到页面上即可显示出来。

5.5.3 更新页面天气信息

视频讲解

将 WXML 页面上所有的临时数据都替换成{{now.属性}}的形式，例如温度是{{now.tmp}}。

WXML(pages/index/index.wxml)代码片段修改如下：

```
1.  <view class='container'>
2.    <!-- 区域 1: 地区选择器 -->
3.    …
4.    <!-- 区域 2: 单行天气信息 -->
5.    <text>{{now.tmp}}℃ {{now.cond_txt}}</text>
6.    <!-- 区域 3: 天气图标 -->
7.    <image src='/images/weather_icon/{{now.cond_code}}.png' mode='widthFix'></image>
8.    <!-- 区域 4: 多行天气信息 -->
9.    <view class='detail'>
10.     <view class='bar'>
11.       <view class='box'>湿度</view>
12.       <view class='box'>气压</view>
13.       <view class='box'>能见度</view>
14.     </view>
15.     <view class='bar'>
16.       <view class='box'>{{now.hum}} %</view>
17.       <view class='box'>{{now.pres}} hPa</view>
18.       <view class='box'>{{now.vis}} km</view>
19.     </view>
20.     <view class='bar'>
21.       <view class='box'>风向</view>
22.       <view class='box'>风速</view>
```

```
23.          <view class='box'>风力</view>
24.        </view>
25.        <view class='bar'>
26.          <view class='box'>{{now.wind_dir}}</view>
27.          <view class='box'>{{now.wind_spd}} km/h</view>
28.          <view class='box'>{{now.wind_sc}} 级</view>
29.        </view>
30.      </view>
31.  </view>
```

运行效果如图 5-20 所示。

需要注意的是,在网速受限的情况下可能不能立刻获取到数据,因此最好在 JS 文件的 data 中为 now 规定初始数据,在获取到实际数据前可以临时显示这些数据。

JS(pages/index/index.js)代码片段修改如下:

```
1.  Page({
2.    /**
3.     * 页面的初始数据
4.     */
5.    data: {
6.      region: ['安徽省', '芜湖市', '镜湖区'],
7.      now:{
8.        tmp:0,
9.        cond_txt:'未知',
10.       cond_code:'999',
11.       hum:0,
12.       pres:0,
13.       vis:0,
14.       wind_dir:0,
15.       wind_spd:0,
16.       wind_sc:0
17.     }
18.   },
19. })
```

在网速受限的状态下,初始数据显示效果如图 5-21 所示。

图 5-20 实况天气数据显示效果

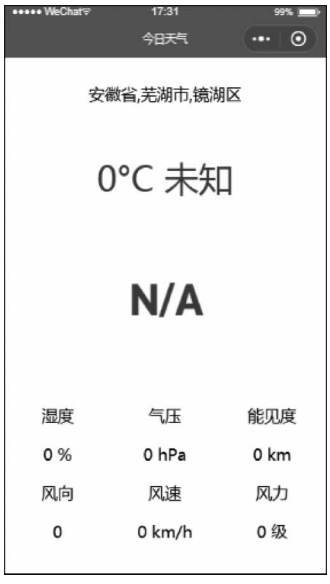

图 5-21 初始数据显示效果

此时项目就全部完成了。

5.6 完整代码展示

app.json 文件的完整代码如下：

```
{
  "pages": [
    "pages/index/index"
  ],
  "window": {
    "navigationBarBackgroundColor": "#3883FA",
    "navigationBarTitleText": "今日天气"
  }
}
```

app.wxss 文件的完整代码如下：

```
/* 背景容器样式 */
.container{
  height: 100vh;                          /* 高度为 100 视窗,写成 100% 无效 */
  display: flex;                          /* flex 布局模型 */
  flex-direction: column;                 /* 垂直布局 */
  align-items: center;                    /* 水平方向居中 */
  justify-content: space-around;          /* 调整内容位置 */
}
```

WXML 文件(pages/index/index.wxml)的完整代码如下：

```
<!-- index.wxml -->
<view class="container">
  <!-- 区域 1: 地区选择器 -->
  <picker mode="region" bindchange="regionChange">
    <view>{{region}}</view>
  </picker>

  <!-- 区域 2: 单行天气信息 -->
  <text>{{now.tmp}}℃ {{now.cond_txt}}</text>

  <!-- 区域 3: 天气图标 -->
  <image src="/images/weather_icon/{{now.cond_code}}.png" mode="widthFix"></image>

  <!-- 区域 4: 多行天气信息 -->
  <view class="detail">
    <view class="bar">
      <view class="box">湿度</view>
      <view class="box">气压</view>
      <view class="box">能见度</view>
    </view>
    <view class="bar">
      <view class="box">{{now.hum}} %</view>
      <view class="box">{{now.pres}} hPa</view>
      <view class="box">{{now.vis}} km</view>
    </view>
```

```
      <view class = 'bar'>
        <view class = 'box'>风向</view>
        <view class = 'box'>风速</view>
        <view class = 'box'>风力</view>
      </view>
      <view class = 'bar'>
        <view class = 'box'>{{now.wind_dir}}</view>
        <view class = 'box'>{{now.wind_spd}} km/h</view>
        <view class = 'box'>{{now.wind_sc}} 级</view>
      </view>
    </view>
</view>
```

WXSS 文件(pages/index/index.wxss)的完整代码如下：

```
/* 文本样式 */
text{
  font-size: 80rpx;
  color:#3C5F81;
}

/* 图标样式 */
image{
  width: 220rpx;
}

/* 区域 4 整体样式 */
.detail{
  width: 100%;
  display: flex;
  flex-direction: column;
}
/* 区域 4 单元行样式 */
.bar{
  display: flex;
  flex-direction: row;
  margin: 20rpx 0;
}
/* 区域 4 单元格样式 */
.box{
  width: 33.3%;
  text-align: center;
}
```

JS 文件(pages/index/index.js)的完整代码如下：

```
Page({
  /**
   * 页面的初始数据
   */
  data: {
    region: ['安徽省', '芜湖市', '镜湖区'],
    now:{
      tmp:0,
      cond_txt:'未知',
      cond_code:'999',
```

```
        hum:0,
        pres:0,
        vis:0,
        wind_dir:0,
        wind_spd:0,
        wind_sc:0
      }
    },
    /**
     * 更新省、市、区信息
     */
    regionChange: function(e) {
      this.setData({region: e.detail.value});
      this.getWeather();              //更新天气
    },
    /**
     * 获取实况天气数据
     */
    getWeather: function() {
      var that = this;
      wx.request({
        url: 'https://free-api.heweather.com/s6/weather/now',
        data:{
          location:that.data.region[1],
          key:'请填入开发者申请的和风天气密钥'     //替换成开发者申请到的 key
        },
        success:function(res){
          that.setData({now:res.data.HeWeather6[0].now});
        }
      })
    },
    /**
     * 生命周期函数--监听页面加载
     */
    onLoad: function(options) {
      this.getWeather();              //更新天气
    }
})
```