王青雅 171250017

石創烽 171250585

梁正川 171250586

张翀 171250587

董新渝 171250588

微信小程序项目说明

———计算机网络作业

南大天气

**1 小组分工**

王青雅：城市的详细天气页面设计

石創烽：小程序基础框架，城市列表界面的美工

梁正川：数据的来源与上传，小程序最终框架

张翀：小程序需求分析，城市搜寻界面的实现

董新渝：选择天气图、详细天气页面背景，完成项目说明报告

**2 功能描述**

为了方便人们用智能手机查看天气预报信息,我们组通过微信小程序开发者工具平台，设计并实现了天气预报微信小程序的开发。小程序主要包括天气预报数据的获取与上传，用户通过微信小程序查询全国城市天气信息和未来天气基本信息。小程序主要功能模块包括**城市列表、城市查询及天气详情显示**。经过测试,结果表明该小程序使用方便,完全能满足人们日常对天气信息的需求。

**3 关键数据逻辑**

在app.js中访问<https://www.tianqiapi.com>获得城市天气数据。

访问方法

get

访问格式

<https://www.tianqiapi.com/api/?version=v1&city=cityName>

如：<https://www.tianqiapi.com/api/?version=v1&city=>北京

返回格式

略（异常庞大的json数据）

本地天气数据

globalData.dataList存储城市对象数组。

每个城市对象格式

{

city: "广州"

cityid: "101280101"

days: [

{wea: "雷阵雨转多云", tem1: "34℃", tem2: "28℃"},

{wea: "雷阵雨", tem1: "34℃", tem2: "28℃"},

{wea: "雷阵雨转大雨", tem1: "34℃", tem2: "26℃"},

{wea: "大雨转大到暴雨", tem1: "32℃", tem2: "25℃"},

{wea: "大到暴雨转大雨", tem1: "30℃", tem2: "25℃"},

{wea: "大雨", tem1: "30℃", tem2: "25℃"}

]

hours: [

{time: "06日08时", tem: "29℃", wea: "小雨"},

{time: "06日11时", tem: "31℃", wea: "多云"},

{time: "06日14时", tem: "32℃", wea: "多云"},

{time: "06日17时", tem: "32℃", wea: "雷阵雨"},

{time: "06日20时", tem: "29℃", wea: "雷阵雨"},

{time: "06日23时", tem: "28℃", wea: "多云"},

{time: "07日02时", tem: "27℃", wea: "多云"},

{time: "07日05时", tem: "27℃", wea: "多云"}

]

img: "<https://timgsa.baidu.com/ti...>"

request\_time: "17:22"

today: {

air:29

humidity:86

tem:"31℃"

tem1:"34℃"

tem2:"27℃"

wea:"雷阵雨转多云"

win:"无持续风向"

}

update\_time: "2019-06-06 11:30:00"

}

云开发

数据库表格式

{

\_id: 57896b495cf145d5092288dc2d62e773

\_openid: o2OVs5VyQ8sLz\_0AqqC3ljs7kYj8 // 用户的openid

cities: ["南京", "北京", "广州"] // 用户保存的城市

}

**4 关键流程说明**

// 城市列表list页面

流程：

1. 通过调用list.js的loadCloudData()，从云端**获取用户之前收藏的城市列表**。loadCloudData()通过调用app.js的requestWeatherByCityName()**请求城市列表中所有城市的天气**。请求结果存至app.js的globalData中，同时本地list.js的data中有一份引用拷贝。
2. 本地list.js中的数据**渲染界面中的城市列表**。每个城市的背景图都能表示该城市的天气，展示该城市天气的概况。否则提示网络环境问题。
3. **左滑**城市可出现**删除键**，删除该城市。
4. **点击加号按钮**，通过调用list.js中的search()跳转到search页面，**添加新的城市**。
5. 添加完毕后，如果网络环境无问题，**新添加的城市加入城市列表中**。
6. **点击城市**，通过调用list.js中的details()跳转到details页面，**展示城市天气的具体情况**。
7. **下拉**可以通过调用list.js中的updateCloudData()**在云端保存收藏的城市**，同时通过调用app.js中的updateAllCities()**刷新列表所有城市的天气预报**。

// 城市查询search页面

流程：

1. 通过调用search.js中的onLoad(),将城市搜索结果初始设定为全部城市（全部城市的数据保存在search.js中的数据列表中）
2. 当用户选择首字母时，通过调用search.js中的bindAZ()，实现根据用户选择首字母，跳转到对应首字母开头城市所在部分。
3. 当用户点击城市时，通过调用search.js中的citySelected(),将该城市名作为参数，实现getApp().globalData.updateWeatherByCityName(cityName)方法的调用，回退到上一界面。
4. 当用户在搜索框输入时，通过调用search.js中的cityInput()，做到根据用户输入实时更新列表选项。每当用户输入一个汉字/字母/拼音时，就会对data中的citys列表遍历，寻找以该字母开头的全部city，并放入cityResultsTemp暂存，全部遍历后，即可得到符合输入条件的城市列表；若无输入，默认为全部城市列表。

// 城市天气详情detail页面

流程：

detail.wxml设计页面总体布局，页面中的数据由detail.js负责调用，调用数据的来源在app.js中的globalData.dataList里。

**5 实验结果说明**

（项目展示）

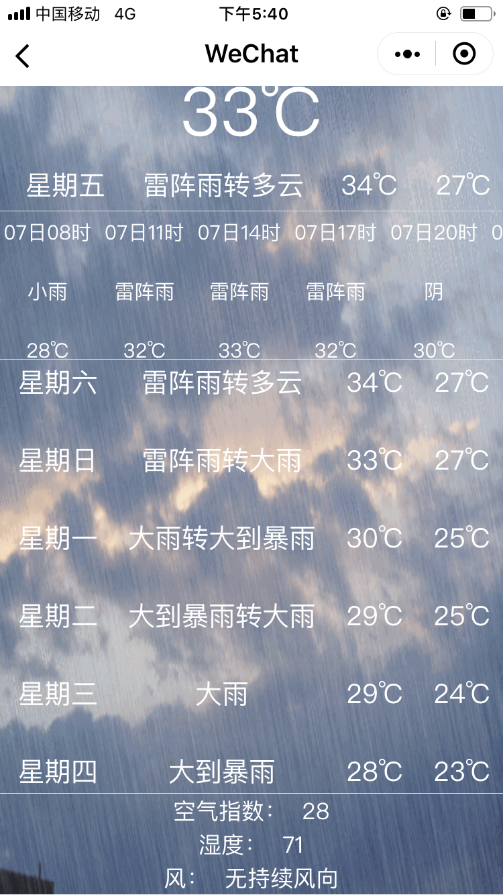
1. 用户点击主页面加号添加需要查看天气的城市



1. 在列表中搜索所需添加的城市，点击完成添加



1. 点击所选城市，查看当地的天气详情，包括当天每间隔三小时的实时天气预报，之后五天的天气预报，以及当天的温度、空气指数、湿度和风向

1. 用户可以左滑已选城市，对列表中城市天气进行删除



1. 用户通过下滑主页面，进行对已选择城市的收藏，并刷新所有城市天气。

