

四川轻化工大学毕业设计（论文）

基于Python的天气质量可视化设计与实现

学 生：王铭亮

学 号：16101010709

专 业：计算机科学与技术

班 级：2016级7 班

指导教师：石睿

四川轻化工大学计算机科学与工程学院

二O二O年六月

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川轻化工大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名： 日期： 年 月 日

关于论文使用授权的说明

本学位论文作者完全了解四川轻化工大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权四川轻化工大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名： 导师签名：

日期： 年 月 日

基于Python的天气质量可视化设计与实现

摘　要

随着信息技术不断发展，人们生活质量逐渐提高，对空气质量的关注越来越多。为满足人们查看天气信息的需求，研究和开发了一个基于Python语言的天气质量可视化软件。软件使用爬虫技术获取四川所有城市和中国各个省会城市的天气质量信息为数据支撑。软件基于Windows平台，以PyCharm为开发工具，采用Python开发语言、Scrapy爬虫框架和PyQt5技术，后台数据库采用MySQL server 2008，实现了该可视化的天气质量查看软件。软件采用模块化和折线图两种不同的信息显示风格，基本满足了现有天气质量信息的查看需求。该软件的设计与实现可以帮助关心空气质量的人们和研究空气质量的研究者，具有一定的经济和社会价值。

关键词：Python，爬虫，Scrapy框架，PyQt5，MySQL

Design and Implementation of Weather Quality Visualization Based on Python

ABSTRACT

With the development of information technology, the quality of life is improving, there is growing concern about air quality. To meet people's need to check weather information, a weather quality visualization software based on Python language is researched and developed. The software USES crawler technology to obtain weather quality information of all cities in Sichuan and provincial capitals of China for data support.The system is based on Windows platform, with PyCharm as the development tool, using Python development language, scrapy crawler framework and PyQt5 technology, Background database adopts MySQL server 2008, realized the visual weather quality view software. The system adopts two different information display styles: modularization and line graph, basically meet the existing weather quality information viewing requirements. The design and implementation of this software can help people who care about air quality and researchers who study air quality, it has certain economic and social value.

Keywords: Python, Crawler, Scrapy Framework, PyQt5, MySQL

目 录

[第一章 绪论 1](#_Toc42411561)

[1.1 研究背景 1](#_Toc42411562)

[1.2 研究目的和意义 1](#_Toc42411563)

[1.3 国内外研究现状 2](#_Toc42411564)

[1.3.1 数据可视化 2](#_Toc42411565)

[1.3.2 研究现状 3](#_Toc42411566)

[1.4 本论文研究的主要内容、难点及技术路线 4](#_Toc42411567)

[1.4.1 主要内容 4](#_Toc42411568)

[1.4.2 重点和难点及解决方法 5](#_Toc42411569)

[1.4.3 技术路线 6](#_Toc42411570)

[第二章 软件开发技术综述 7](#_Toc42411571)

[2.1 Python语言 7](#_Toc42411572)

[2.2 爬虫技术 7](#_Toc42411573)

[2.2.1 爬虫设计分析 8](#_Toc42411574)

[2.2.2 Scrapy框架 9](#_Toc42411575)

[2.3 MySQL数据库 11](#_Toc42411576)

[2.4 PyQt5框架与信号槽 11](#_Toc42411577)

[2.5 传输协议与Socket技术 12](#_Toc42411578)

[2.5.1 TCP与HTTP传输协议 12](#_Toc42411579)

[2.5.2 Socket套接字 13](#_Toc42411580)

[2.6 本章总结 13](#_Toc42411581)

[第三章 需求分析 14](#_Toc42411582)

[3.1 可行性分析 14](#_Toc42411583)

[3.1.1 技术可行性分析 15](#_Toc42411584)

[3.1.2 运行可行性分析 15](#_Toc42411585)

[3.2 系统分析 15](#_Toc42411586)

[3.3 功能需求分析 15](#_Toc42411587)

[3.4 非功能需求分析 16](#_Toc42411588)

[3.5 本章小结 16](#_Toc42411589)

[第四章 系统总体设计 17](#_Toc42411590)

[4.1 系统设计原则 17](#_Toc42411591)

[4.2 系统设计要求 17](#_Toc42411592)

[4.2.1 设计目标 18](#_Toc42411593)

[4.2.2 开发及运行环境 18](#_Toc42411594)

[4.3 软件功能结构 18](#_Toc42411595)

[4.4 系统功能概述 19](#_Toc42411596)

[4.5 数据库设计 19](#_Toc42411597)

[4.6 查询模块 23](#_Toc42411598)

[4.7 本章小结 23](#_Toc42411599)

[第五章 系统实现 24](#_Toc42411600)

[5.1 气象和空气质量信息提取 24](#_Toc42411601)

[5.1.1 选择及分析数据 24](#_Toc42411602)

[5.1.2 构造URL访问接口 25](#_Toc42411603)

[5.2 数据的存储 25](#_Toc42411604)

[5.3 Socket数据传输 27](#_Toc42411605)

[5.3.1 Client客户端 28](#_Toc42411606)

[5.3.2 Server服务器 28](#_Toc42411607)

[5.4 可视化编程 29](#_Toc42411608)

[5.5 软件主要功能 30](#_Toc42411609)

[5.5.1 软件主要界面 30](#_Toc42411610)

[5.5.2 最小化至系统托盘 32](#_Toc42411611)

[5.5.3 服务器响应 32](#_Toc42411612)

[5.5.4 自动化爬虫 33](#_Toc42411613)

[5.6 小结 33](#_Toc42411614)

[第六章 系统测试 34](#_Toc42411615)

[6.1 测试的目的与目标 34](#_Toc42411616)

[6.2 测试方法 34](#_Toc42411617)

[6.3 测试用例 35](#_Toc42411618)

[6.4 测试结论 36](#_Toc42411619)

[第七章 结论与展望 37](#_Toc42411620)

[7.1 结论 37](#_Toc42411621)

[7.2 展望 37](#_Toc42411622)

[致　谢 39](#_Toc42411623)

[参考文献 40](#_Toc42411624)

[附录 42](#_Toc42411625)

[附录A:软件使用说明书 42](#_Toc42411626)

[附录B:主要源程序 44](#_Toc42411627)

# 第一章 绪论

## 1.1 研究背景

气象数据是开展天气预警预报、气候预估及各类气象服务、科学研究的基础，是推动气象科学发展的原动力。大量而繁杂的数据更好的被应用起来，预报结果更形象、直观的令大众接受理解，都离不开气象可视化的功劳。可视化的概念源自计算机，并且与计算机捆绑在一起。它是科学研究和服务的辅助工具。随着计算机技术的不断发展，越来越多的可视化开发工具被用在气象领域，如GrADS、NCL、IDL等，许多气象数据可视化方面的专家学者对这些工具在数据处理上的应用开展了专门的研究。

Python作为一种计算机程序设计语言。最初被设计用于编写自动化脚本，但是由于它本身具有可扩展的特性，可以加载C/C++模块来进行功能扩展。因此到现在为止Python加载了许多包含GUI控件集的模块来提供图形化程序开发。而PyQt就是Qt为Python提供的图形化程序开发扩展模块，其制图功能不亚于Matlab、Grads、NCL等专业绘图工具，并且借助于Python强大的功能和简洁的语法，使其在编程方面要胜于以上绘图工具。

## 1.2 研究目的和意义

在这个互联网和大数据的时代，各种数据充斥着人们的生活中，如何充分利用这些数据是许多人长期以来一直关注的问题。随着嵌入式和计算机等科技高速发展，数据采集能力不断提高，涉及AQI、气温、湿度、气压、风向、风速等采集数据无限扩大，且采集时间不断缩短，从几十小时到十几分钟，甚至几分钟。我国目前有超过5万个地面观测气象站，还有天空、卫星等观测点，可以采集到海量的气象数据。西方经济学中的德尔菲气象规律明确指出在气象信息研究上的投入与产出比为1比98，也就是说投资1元可以赚到98元。

在国外，气象服务大多商业化，英国气象服务年产值高达2600亿美元，美国1600亿，日本100亿，然而中国只有6亿美元[1]，由此可见，专业的气象服务在中国有着巨大的前景。如今，气象数据已不再是气象部门内部的机密数据，气象数据已经被中国气象网作为了一种云服务，开放了气象数据文件下载供个人和商业研究和使用，中国气象局公共气象服务中心与阿里云达成战略合作，共同挖掘气象大数据的深层价值。

因此，对于如何直观、清晰的展示气象数据显得极其重要。目前存在的许多气象数据可视化平台有基于桌面应用的，也有基于web应用的，但总体来看这些可视化数据平台大多是提供给专业气象人士使用的，平台上使用的色斑图、雷达图等图形只有专业人士能够读懂，对于广大非专业的公众来说，他们并不一定能看懂那些数据图形。如今，数据可视化技术发展迅猛，人们对数据掌控的需求日益增长，如果公众可以通过交互的方式，按需在软件上浏览直观的二维趋势图或者模块式显示信息的形式来了解天气预报信息，还可以了解过去和未来的天气变化趋势，那么将更好地满足公众对天气预报的需求，大大地提升人们对于天气的了解和掌控能力。

## 1.3 国内外研究现状

### 1.3.1 数据可视化

数据可视化的发展要追溯到史前时代，那个时候可视化作品也许是在一块沙地上，或是刻在岩石上，古老的地形图中显示的地址信息竟然已经达到了“精确”的程度。数据可视化发展到现在，已经完全不是只停留在纸上绘画，而是结合计算机技术，开发出更加个性化的、全方位的数据可视化产品[2]。目前，国内外关于数据可视化的研究都获得了很大的研究成果，数据可视化技术日新月异，应用范围越来越广，其带来的影响也逐渐凸显。研究可视化技术的行业可分为世界高校、政府和各大企业，每年都会开展许多数据可视化国际会议。

国外研究现状目前，国外有许多有关数据可视化的书籍，如《可视化数据》、《大数据可视：重构智慧社会》等，书中详细解析了数据获取方式、数据可视化的分类、数据可视化的设计过程、数据可视化技术、工具选取和数据可视化的应用[3]。许多知名企业（例如：Miscrosoft、IBM、SAS）都在数据可视化方面有显著的成果，例如著名的OpenGL、directX和highcharts等[4]。除此之外，还有：

（1） SAS公司的SASRVisual BI。这是一款商务智能软件，该软件可以展现大量数据的动态交换效果，推动了对商务有关数据的研究以及一系列的应用发展。

（2） Gephi。这是一款进行社会图谱数据可视化分析的软件，可以构建动态分层的数据图表，是一个可视化网络探索平台。

国外对于数据可视化的研究不仅只是展示现有数据，而且在不同领域还有不少结合了大数据，运用了数据挖掘技术，数据分析，利用数据可视化展示其分析预测的结果，推动了各个领域更好的监管和防范风险[4]。

国内在数据可视化研究方面也有一定的成果。许多互联网企业都推出了自己的可视化平台、云服务可视化系统、数据智能分析平台，致力于把数据可视化技术打造成一种服务，给广大用户提供按需服务，使技术资源充分利用[5]。以下是一些国内的数据可视化成果：

（1） 阿里云发布的大数据平台“数加”。它是全球首个一站式大数据平台，它提供的产品包括移动数据分析、DataV可视化、规则引擎、推荐引擎、BI报表、应用托管、郡县图治。

（2） 360大数据平台可视化产品。该平台提供的产品包括“骗子地图”和“360星图”，人们通过“骗子地图”看见的数据，真切的揭示了互联网的骗局[6]。

### 1.3.2 研究现状

在现实世界，许多现象都具有动态性、空间性，只有通过特定的环境或平台才能将这些连续且变化的海量数据直观地展现出来，气象数据就是一个典型的现象数据，目前，如何更加直观的可视化表达海量的气象数据来满足公众工作和生活的需要，是各个行业乃至气象研究平台都在一直关注的问题。

通过可视化平台，可以将复杂的海量的气象数据变得具有可读性、观赏性和判断性，将专业人士才懂的专业词汇转换成通俗易懂的图形，专业的天气现象转换成直观容易理解的图片或文字形式，已达到使各个领域的用户都能充分获取、理解气象信息的目的，从而提升气象服务传播的效果，增强公众对气象信息知识的了解。

国外研究现状：

为了在气象领域广泛发展GIS技术，在2001年，科学技术领域欧洲协作研究计划开始了COST719计划，由18个欧洲国家参加研究，其目标为：密切合作GIS产业，建立GIS与气象数据的接口，对气象数据的实用性和可获得性进行评估，鼓励GIS技术在气象研究中的应用，使国家气象服务机构、研究组织和GIS行业密切联系[7]。

Dark Sky 实时天气预报网由AdamGrossman 和JackTurner创办，打开Dark Sky 网站，Dark Sky天气信息如图1-1所示，用户在一个很漂亮的地图上浏览未来15分钟天气的变化。

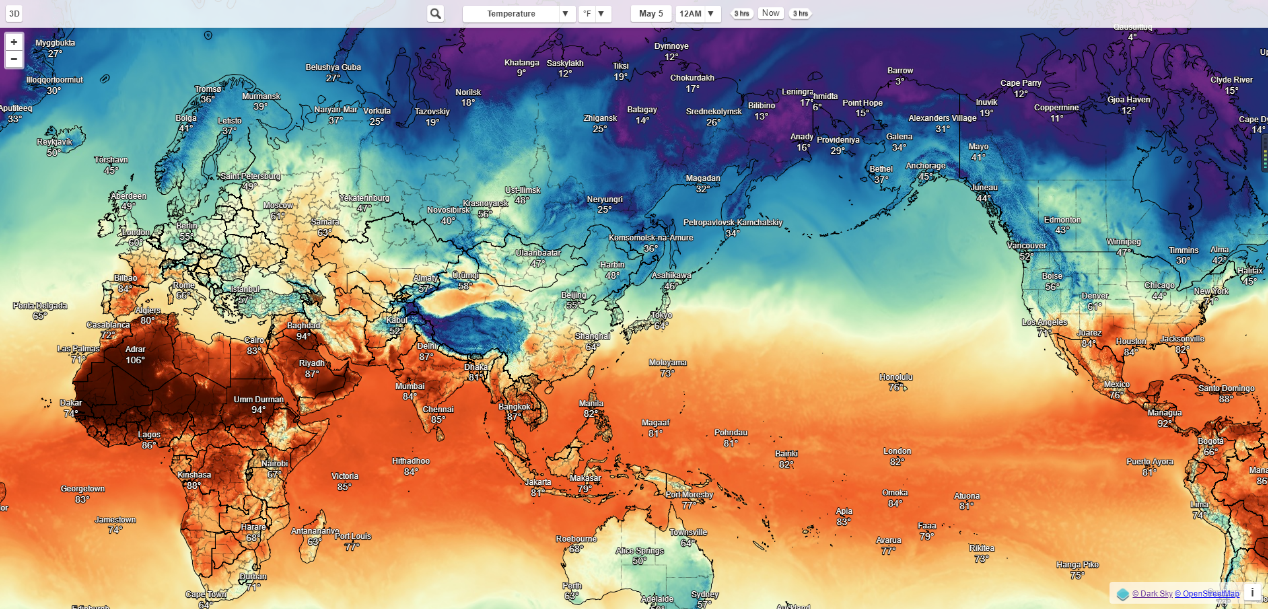


图1-1 Dark Sky天气信息

国内研究现状：

2006年11月7日召开的MICAPS－V3.0系统（新一代天气预报人机交互处理系统）联合开发研讨会，以中国气象局国家气象中心为设计开发核心，该系统可以实时处理数据，并对数据进行分析，提升了天气预报可视化水平，它拥有一套通俗易懂的、直观的气象数据可视化图表，有效地提高了专业气象人员的工作效率[7]。安徽财经大学管理科学与工程学院利用Python爬虫技术，选用2014—2018年合肥市空气质量数据。分别从空气质量指数、质量等级、空气质量预测三个方面对合肥市空气质量进行可视化分析和总结，揭示合肥市空气质量特征和未来空气质量发展趋势。

## 1.4 本论文研究的主要内容、难点及技术路线

### 1.4.1 主要内容

本论文利用设计一个基于Python的可视化天气质量（包括气象与空气质量信息）软件，该系统作为地面天气质量观测数据服务的子系统，采用C/S构架进行开发，结合了Python爬虫技术、Pyqt5技术、Pyecharts网页绘图技术，全方位更直观的对天气质量进行可视化，主要研究内容包括以下几项内容：

（1） Scrapy爬虫框架；

（2） Pyqt5可视化；

（3） MySQL数据存储；

（4） Socket套接字技术；

（5） Pyecharts网页绘图技术。

### 1.4.2 重点和难点及解决方法

重点：

在Linux环境下搭建爬虫服务器，使用Scrapy爬虫框架对气象数据和空气质量数据接口进行爬取，将实时的气象和空气质量数据及气象与空气质量近几天预报的数据进行格式化后存入MySQL数据库，使用Socket套接字获取客户端发来的需要查找的城市名请求，进行查询后将准备好的数据通过接口进行返回，在客户端进行显示。其中会使用到二维趋势图的展示方式，使用到了Pyecharts[8]，通过将数据进行整合。其中会标注最大值与最小值使用户一目了然。城市空气质量日报折线图如图1-2所示。

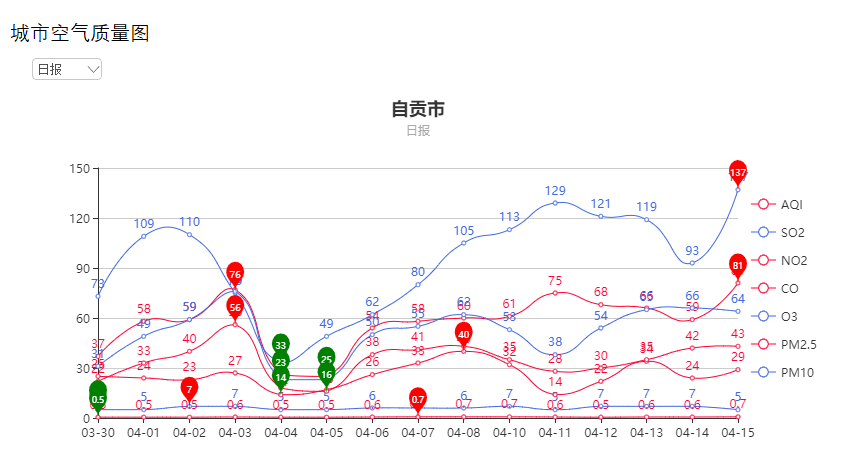


图1-2 城市空气质量日报折线图

难点及解决方法：

（1） 爬虫请求失败：使用爬虫最怕请求服务失败，此时需要使用代理ip池或者伪装成各种浏览器再或者使用不同的请求头。

（2） 构造实时空气质量接口的URL：获取实时数据时需要将系统被格式化的时间转化为时间戳的形式与URL进行构造获得一个实时数据接口，然后再获取数据。

（3） 自动运行脚本：在Linux系统中定时运行爬虫脚本时，使用crontab命令。其中设置运行时间是关键，\*(分钟) \*(小时) \*(日) \*(月) \*(星期) command(执行的命令)。

（4） 服务器监听客户端是否复用或断线：使用Socket监听客户端是否连接时会出现地址复用问题。此时需要使用setsockopt()函数允许地址复用；当客户端断开连接时服务器也会出现错误，此时服务器监听时需要设置错误处理机制进行判断。

（5） 在滚动栏动态生成列表：使用PyQt5动态生成城市列表，需要使用for循环，而且要实现在单击列表中的按钮时做出对应操作，此功能设计时需要使用按钮信号槽的单击事件。更重要的是需要使用sender()函数用来记录发出信号的对象，因为实在for循环里生成的列表，如果使用普通的text()函数监听发出的信号会出现bug。

（6） 自定义标题栏和关闭按钮：当隐藏PyQt5自带的标题栏后，便无法对窗口进行移动，此时需要自定义移动窗体的类即class。使用QmouseEvent. globalPos() 和pos()函数获取鼠标在窗体的相对位置，使用move()函数更改窗口位置。

（7） 系统托盘显示与自动关闭小图标：为了使用户方便打开软件查找天气状况，且又不影响用户其他操作，设计了将窗口最小化到系统托盘，但退出程序后遇到了在系统托盘的图标无法自动关闭的问题。

### 1.4.3 技术路线

本软件设计基于Linux和Windows系统，使用Python语言进行服务器和客户端设计，使用Linux做服务器是因为Linux环境下的编程有更好的兼容性。之所以客户端在Windows系统下进行设计是考虑到现大多数人群使用的办公电脑系统是Windows系统。两种系统也符合现在大多数编程人员选择的习惯。系统设计流程图如图1-3所示。

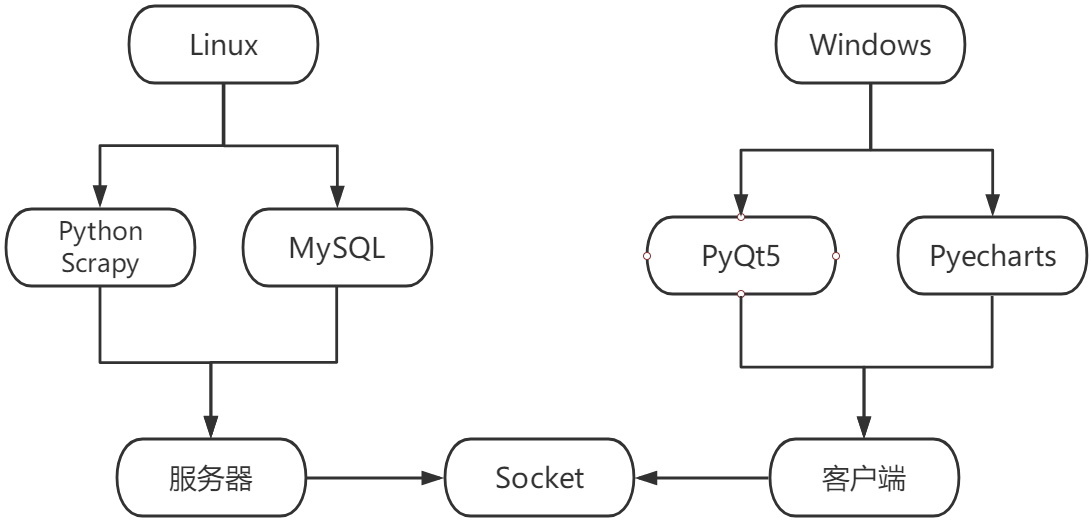


图1-3 系统设计流程图

# 第二章 软件开发技术综述

## 2.1 Python语言

Python是一门开源通用型编程语言,有着良好的社区支持,数据库模块规范齐全[9]，由于其自身特点,在很多领域都非常优秀,是一门真正的全栈语言。对于未来人工智能领域的发展，Python语言会继续发挥其不可代替的作用，推进信息时代的持续发展[10]。Python是一种代表简单主义思想的语言，是一个高层次的结合了解释性、编译性、互动性和面向对象的脚本语言。阅读一个良好的Python程序就感觉像是在读英语一样，尽管这个英语的要求非常严格，这种伪代码本质是它最大的优点之一。它使你能够专注于解决问题而不是去搞明白语言本身。具有丰富和强大的库。它常被昵称为胶水语言，能够把用其他语言制作的各种模块（尤其是C/C++）很轻松地联结在一起。常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。

## 2.2 爬虫技术

网络爬虫是一个自动提取网页的程序，它为搜索引擎从万维网上下载网页，是搜索引擎的重要组成。传统爬虫从一个或若干初始网页的URL开始，获得初始网页上的URL，在抓取网页的过程中，不断从当前页面上抽取新的URL放入队列,直到满足系统的一定停止条件。聚焦爬虫的工作流程较为复杂，需要根据一定的网页分析算法过滤与主题无关的链接，保留有用的链接并将其放入等待抓取的URL队列。然后，它将根据一定的搜索策略从队列中选择下一步要抓取的网页URL，并重复上述过程，直到达到系统的某一条件时停止。另外，所有被爬虫抓取的网页将会被系统存贮，进行一定的分析、过滤，并建立索引，以便之后的查询和检索；对于聚焦爬虫来说，这一过程所得到的分析结果还可能对以后的抓取过程给出反馈和指导。

相对于通用网络爬虫，爬虫还需要解决三个主要问题：

(1) 对抓取目标的描述或定义；

(2) 对网页或数据的分析与过滤；

(3) 对URL的搜索策略。

### 2.2.1 爬虫设计分析

当开始使用爬虫时需要对爬虫进行设计，这样可以有效地使用爬虫。爬虫设计流程如图2-1所示。

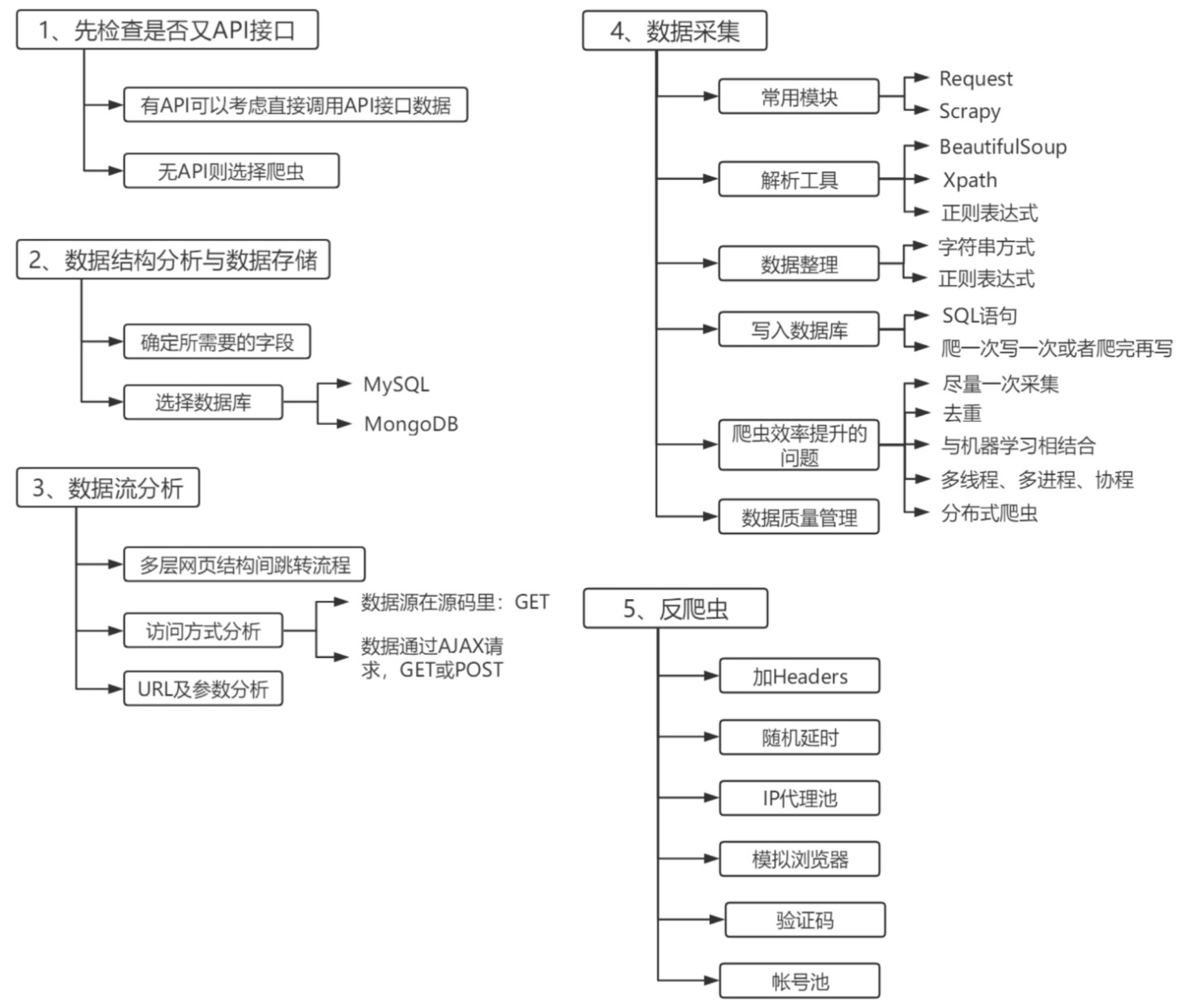


图2-1 爬虫设计流程

1. 先检查是否有API

API是网站官方提供的数据接口，如果通过调用API采集数据，则相当于在网站允许的范围内采集，这样既不会有道德法律风险，也没有网站故意设置的障碍。

1. 数据结构分析和数据存储

爬虫需求要十分清晰，具体表现为需要哪些字段，这些字段可以是网页上现有的，也可以是根据网页上现有的字段进一步计算的，这些字段如何构建表，多张表如何连接等。

1. 数据流分析

对于要批量爬取的网页，往上一层，看它的入口在哪里；这个是根据采集范围来确定入口，比如若只想爬一个地区的数据，那从该地区的主页切入即可；但若想爬全国数据，则应更往上一层，从全国的入口切入。

1. 数据采集

Scrapy是一个成熟的爬虫框架，使用方便爬取流程清晰。Requests是一个比原生的urllib包更简洁强大的包，适合作定制化的爬虫功能。

1. 解析工具

BeautifulSoup比较简单，支持Xpath和CSSSelector两种途径。

正则表达式很强大，但构造起来有点复杂，需要专门去学习。因为下载下来的源码格式就是字符串，所以正则表达式可以大显身手，而且处理速度很快。

1. 数据整理

可以使数据看起来更干净，插入数据库更方便。

1. 写入数据库

写入数据库有两种方法，一种是通过Pandas的DataFrame自带的to\_sql()方法；另一种是利用数据库引擎来执行SQL语句。

1. 爬虫效率提升

尽量减少访问次数。值得深思的一点就是，是不是所有的数据都需要爬。

大量爬虫是一个IO阻塞的任务，因此采用多进程、多线程或者协程的并发方式可以有效地提高整理速度。

1. 数据质量管理

try...except是Python中常用的异常诊断语句，在爬虫中也可充分应用。一方面，避免爬取该字段的语句出错；另一方面，该语句可以跳过中间出现的各种自己产生或者网站产生的错误，保证爬虫整体的持续进行。

1. 反爬虫技术

反爬虫技术有以下几种：加上headers；随机延时；IP代理池；模拟浏览器；验证码。

### 2.2.2 Scrapy框架

Scrapy是适用于Python的一个快速、高层次的屏幕抓取和web抓取框架，用于抓取web站点并从页面中提取结构化的数据。Scrapy用途广泛，可以用于数据挖掘、监测和自动化测试[11]。它吸引人的地方在于它是一个框架，任何人都可以根据需求方便的修改。它也提供了多种类型爬虫的基类，如BaseSpider、sitemap爬虫等，最新版本又提供了web2.0爬虫的支持[12]。Scrapy框架的流程[13]。Scrapy框架流程如图2-2所示。

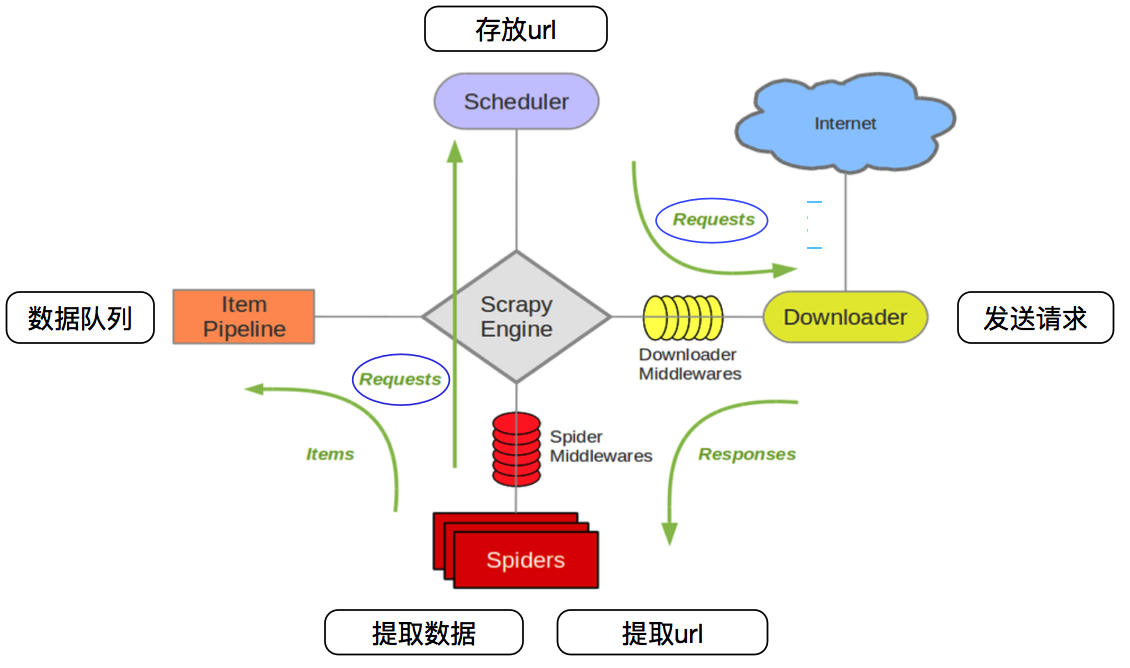


图2-2 Scrapy框架流程

• Scrapy Engine(引擎)：负责Spider、ItemPipeline、Downloader、Scheduler中间的通讯，信号、数据传递等。

• Scheduler(调度器)：它负责接受引擎发送过来的Request请求，并按照一定的方式进行整理排列，入队，当引擎需要时，交还给引擎。

• Downloader（下载器）：负责下载Scrapy Engine(引擎)发送的所有Requests请求，并将其获取到的Responses交还给Scrapy Engine(引擎)，由引擎交给Spider来处理。

• Spider（爬虫）：它负责处理所有Responses,从中分析提取数据，获取Item字段需要的数据，并将需要跟进的URL提交给引擎，再次进入Scheduler(调度器)。

• Item Pipeline(管道)：它负责处理Spider中获取到的Item，并进行后期处理（详细分析、过滤、存储等）的地方。

• Downloader Middlewares（下载中间件）：你可以当作是一个可以自定义扩展下载功能的组件。

• Spider Middlewares（Spider中间件）：你可以理解为是一个可以自定扩展和操作引擎和Spider中间通信的功能组件，比如，进入Spider的Responses和从Spider出去的Requests。

## 2.3 MySQL数据库

MySQL是一个中、小关系型数据库管理系统，由瑞典MySQL AB公司开发，目前属于Oracle公司旗下产品。MySQL是最流行的关系型数据库管理系统之一，是一种关联数据库管理系统[14]，关联数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。Mysql的复杂性很低，但是仍然提供广泛的数据库功能和选项。与其他数据库相比，比如Oracle，公司可以为开发人员立即配置数据库，例行任务非常简单，而且操作简单，最重要的是可以降低总拥有成本，这些有利条件使得越来越多的公司选择MySQL作为数据库[15]。

MySQL是小型关系型数据库管理系统的代表，MySQL具有占用体积小，运行速度快，系统花费低和源码开放的特点；数据库不仅是用来存放系统所需的相关数据，还是整个系统前后端交互的枢纽。作为数据库入门级人员，选择了轻巧方便，兼容性较好的MySQL数据库。选择该数据库也是从系统成本考虑选择的结果。MySQL所使用的SQL语言是用于访问数据库的最常用标准化语言。

## 2.4 PyQt5框架与信号槽

PyQt是Qt框架，Python语言实现，由Riverbank Computing开发，是最强大的GUI库之一。PyQt提供了一个设计良好的窗口控件集合，每一个PyQt控件都对应一个Qt控件，因此PyQt的API接口与Qt的API接口很接近。 PyQt5本身拥有超过620个类和6000函数及方法。在可以运行于多个平台，包括：Unix, Windows, Mac OS[16]。它有以下几个优势：

1. 优良的跨平台特性

Qt支持下列操作系统: Microsoft Windows 95/98， Microsoft Windows NT， Linux， Solaris， Digital UNIX (OSF/1， Tru64)， FreeBSD， OS390，QNX 等。

1. 面向对象

Qt的良好封装机制使得Qt的模块化程度非常高，可重用性较好，对于用户开发来说是非常方便的。Qt提供了一种称为signals/slots的安全类型来替代callback，这使得各个元件之间的协同工作变得十分简单。

1. 丰富的API

Qt包括多达250个以上的C++类，还提供基于模板的collections，serialization，file，I/Odevice，directory management，date/time类。甚至还包括正则表达式的处理功能。

1. 支持2D/3D图形渲染，支持OpenGL。
2. 大量的开发文档。
3. XML支持。

信号和槽机制是PyQt5定义的一种独特的通信机制，也是PyQt5区别于其他GUI控件集的重要特征。在其他GUI控件集中，窗口中的控件Widget一般都采用回调函数对相应的事件进行处理，回调函数通常是一个指向某个函数的指针，该函数将在指定的事件发生时被调用。PyQt5用信号和槽取代了这些凌乱的函数指针，从而使控件和事件之间的通信变得更为简洁明了。

在PyQt5中，所有从QObject或其子类(如QWidget)派生的类都能够包含信号和槽。当某个对象的状态发生改变时，相应的信号就由该对象发射（emit）出去，这是该对象所要完成的全部事情，它并不需要知道另一端是谁在接收这个信号，从而实现了真正意义上的信息封装。PyQt5中槽的作用则是负责接收信号,并对该信号作相应的处理，一个槽并不需要知道是否有信号与之相连。

## 2.5 传输协议与Socket技术

设计选择C/S的模式，所以会涉及数据传输的问题，本章对两种协议，即TCP和HTTP协议进行介绍，选择合适的传输协议进行C/S模式的设计。

### 2.5.1 TCP与HTTP传输协议

• TCP协议

传输控制协议（Transmission Control Protocol），建立起一个TCP连接需要经过“三次握手”：

握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP连接一旦建立，在通信双方中的任何一方主动关闭连接之前，TCP 连接都将被一直保持下去。断开连接时服务器和客户端均可以主动发起断开TCP连接的请求，断开过程需要经过“四次握手” [17]。

• HTTP协议

即超文本传送协议（Hypertext Transfer Protocol），是Web联网的基础，也是手机联网常用的协议之一，HTTP协议是建立在TCP协议之上的一种应用。

HTTP连接最显著的特点是客户端发送的每次请求都需要服务器回送响应，在请求结束后，会主动释放连接。从建立连接到关闭连接的过程称为“一次连接”。

由于HTTP在每次请求结束后都会主动释放连接，因此HTTP连接是一种“短连接”，要保持客户端程序的在线状态，需要不断地向服务器发起连接请求[18]。

总的来说TCP和HTTP是地位对等的协议，虽然TCP是传输层，而HTTP是应用层，但HTTP是要基于TCP连接基础上的，也就是说TCP就是单纯建立连接，不涉及任何我们需要请求的实际数据，简单的传输。而HTTP是用来收发数据，即实际应用上来的。

### 2.5.2 Socket套接字

• Socket连接与TCP连接

创建Socket连接时，可以指定使用的传输层协议，Socket可以支持不同的传输层协议（TCP或UDP），当使用TCP协议进行连接时，该Socket连接就是一个TCP连接。

• Socket连接与HTTP连接

HTTP连接使用的是“请求—响应”的方式，不仅在请求时需要先建立连接，而且需要客户端向服务器发出请求后，服务器端才能恢复数据。

很多情况下，需要服务器端主动向客户端推送数据，保持客户端与服务器数据的实时与同步。此时若双方建立的是Socket连接，服务器就可以直接将数据传送给客户端；若双方建立的是HTTP连接，则服务器需要等到客户端发送一次请求后才能将数据传回给客户端，因此，客户端定时向服务器端发送连接请求，不仅可以保持在线，同时也是在“询问”服务器是否有新的数据，如果有就将数据传给客户端[19]。

## 2.6 本章总结

本章主要介绍了Python语言、爬虫技术、Scrapy框架、MySQL数据、传输协议与Socket技术、PyQt5框架，详细分析了各个技术的相关特点，为下面的软件设计打下了良好的技术基础。

# 第三章 需求分析

针对系统需求内容的分析是开发设计工作的重要基础与前提，在需求分析过程中，应当始终紧紧围绕用户的需求[20]。在对系统进行详细梳理的基础上，需要明确完成各项内容。系统的需求分析是开发项目中很重要的基础，是项目开发者与使用者相互沟通的结果，是一个项目是否能够成功的前提，也是评判一个系统优劣的重要依据。

## 3.1 可行性分析

随着科技不断发展，生活节奏越来越快，对生活质量要求逐渐增高，对天气质量的关注也逐渐增多，原有的天气质量查询网站无法满足人们查询天气信息的要求，特别是在PC端，更是没有一个稳定简洁的天气质量查询软件，导致天气信息发展的很缓慢。

如何便捷地查询气象信息和空气质量信息和他们变化的趋势，一直是研究人员和开发人员的一大难题。

1. 在何种环境下进行查询

查询途径很多，人们往往第一反应是打开浏览器使用百度，但是气象信息网站和空气质量信息网站繁多，选择哪一个网站才能准确获取想要的信息，又成了一个问题。

1. 气象信息和空气质量信息往往是在不同网站展示

一般用户查询数据要么查询气象信息或者空气质量信息，当查询其中一个信息后会忽略另一个信息，这样两种信息很难得到用户的长时间关注和反馈，从而让气象信息和空气质量信息相对于其他国家发展的很慢。

1. 数据以数字形式显示

信息数据以冰冷的数字进行显示，表示不明确使用户或研究人员无法对以往数据做对比，判断往后几天乃至几周的气象变化趋势。

综合以上问题，本设计从方便用户和研究人员查询数据出发，结合气象和空气质量信息，将数字化的数据以二维趋势图标进行展示，解决已经发现的问题。用对比的方法进行气象和空气质量信息的研究，探索数据的查询和展示问题，为气象和空气质量信息发展提供借鉴行思路。

### 3.1.1 技术可行性分析

系统实现基于Python语言与PyQt5框架，数据采集采用Scrapy框架，数据库存储采用MySQL。PyQt5不仅集成了Qt的面向对象和跨平台的优点，而且软件设计基于Python，这样拥有了更好的兼容性。Scrapy爬虫框架不仅提供了简洁地爬取接口，更将数据处理模块单独分出来，可供后期统一对数据进行管理。MySQL数据库的选择因为兼容性较好更方便了开发的流程，使软件的数交流更加快捷轻便。因此在技术上可行。

### 3.1.2 运行可行性分析

目前办公方式都是选择电脑，只需在PC端安装上软件打开便可运行，且对硬件和内存的配置要求也不高，在使用上没有任何问题，用户无需经过培训，只需根据软件主界面的提示进行操作计科，因此，系统在投入使用中阻力较小，基本可行。

## 3.2 系统分析

基于Python的天气质量可视化软件的开发主要包括PC端的使用界面、后台天气信息数据库和数据传输服务器，后台数据库的建立及维护要求数据具有完整性、一致性和实时更新性，是数据更新迅速且完整的数据库。对于前端可视化的开发，提供了简单快捷的操作功能。

## 3.3 功能需求分析

在本次开发过程当中，天气质量可视化软件功能要求包括几个方面，当天实时气象信息查询、气象预报信息查询、空气质量预报信息查询、空气质量小时报趋势图查询、空气质量日报趋势图查询。

查询方式分为两种，一种是使用搜索栏进行搜索，另一种是使用城市列表导航按钮进行查询，信息以二维模块的形式展示。天气质量信息及预报查询用例图如图3-1所示。

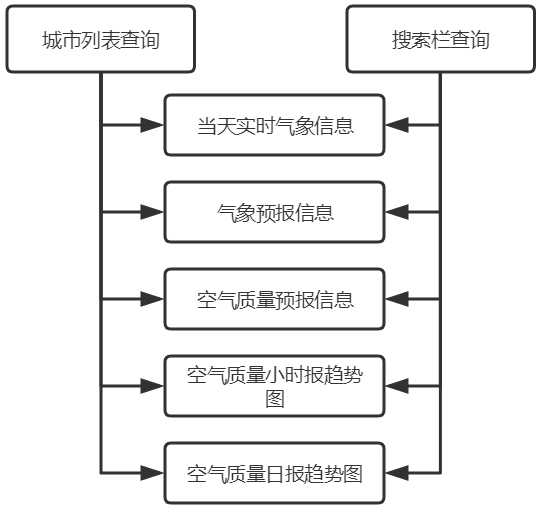


图3-1 天气质量信息及预报查询用例图

## 3.4 非功能需求分析

软件的外部需求在开发的整个过程中应用十分广泛,天气质量信息查新软件涉及的主要外部需求如下：

1. 简单性需求：该系统在操作方面非常容易上手，界面良好。只有这样才能实现节省用户查询时间，美观大气的设计更受用户的青睐。
2. 可扩展性需求：软件的扩展性很重要，本软件设计了用户登录接口，可以记录用户习惯和快速定位用户所在城市。还设有开机自启动设置，方便用户开机就可以查看到当地天气信息。

## 3.5 本章小结

本章主要是对软件进行需求分析。软件的需求分析首先从可行性分析开始，确定了系统的可行性。接着确定软件的总体需求。然后对软件的功能性需求进行分析，确定软件所要实现的功能。最后通过非功能性需求分析确定了软件的可行性。这些分析将为软件的设计提供丰富的参考和依据。

# 第四章 系统总体设计

在对项目整体分析之后，对项目的结构及功能要有清晰合理的架构，要满足软件的功能需求。根据需求分析及系统规格说明，从项目的整体结构入手，对软件需要实现的功能做出合理化设计，在软件功能的说明中，应当将功能限定权限的类别明确出来，确保软件使用效果得以被充分发挥出来[21]。在后台数据库的支持下，完成客户端查询天气信息功能。整个项目从功能要求出发，从项目的整体设计结构开始，层层递进，逐步设计出完善的功能。

## 4.1 系统设计原则

在天气质量可视化界面的编码过程中，通过规范的、清晰的布局方式能够反映系统的结构，能够更好的完善功能结构，能够节省在设计过程中占用的时间，为后续操作提供更加富足的时间，以便于降低效益成本。以下是对各项设计原则的介绍：

1. 高效性

在软件的系统设计方面，要求设计合理，减少系统硬件资源的耗用，同时也要根据用户需求，是软件能够实现跨平台使用天气质量可视化软件。

1. 可靠性

在系统运行过程中，不可忽视可靠性的问题，由全局出发提高系统可靠性，设计方案需要择优选择，减少系统故障的发生。

1. 可维护性

软件能否正常运行直接由自身可维护性决定，在软件运行期间难免会遇到各种问题，故障是随时随地存在的，所以一个系统必须具有可维护性，也便于降低故障带来的损失。

## 4.2 系统设计要求

考虑到软件的总体设计，整个系统的设计应简单快捷；操作应该具有人性化的特点，数据展示需要做到简洁明了，查询操作多样化且简介；编码工具和方式要统一，防止因版本原因导致不必要的隐患出现。

### 4.2.1 设计目标

利用天气质量可视化软件可以使用户能够简单快速地完成查询各个城市的天气信息，减少网页查询的繁琐步骤。大大提高人们对空气质量的关注，以及提高研究人员的工作效率。本软件的具体目标如下：

1. 软件页面美观大方，功能操作灵活方便，数据显示明了且完整；
2. 城市信息可通过搜索框实现查询，保证用户完美体验；
3. 与数据库实现数据的完美对接，使后台数据完美显示到软件界面上；
4. 后台数据库的管理要最大限度地保证其版本的正常使用和规避人员操作所带来的影响，主要表现在爬取接口数据时候无法获取数的情况，使其兼容性、维护性和操作性均稳定可靠；
5. 爬虫流程需要简洁且高效，做到自动化运行。

### 4.2.2 开发及运行环境

天气质量可视化软件的开发环境为PyQt5和Pyecharts，运行环境为Windows 10。服务器开发环境为Scrapy和MySQL，运行环境为Linux，数据库开发选择轻巧方便兼容性强大的MySQL 2008，服务器传输选择TCP。

## 4.3 软件功能结构

软件的功能结构包括：城市天气预报、空气质量图表、设置、城市天气信息搜索、用户登录、城市实时气象；数据库的数据主要包括：城市气象信息表、城市气象信息预报表、城市空气质量时报信息表、城市空气质量小时报信息、城市空气质量预报。程序功能结构如图4-1所示。

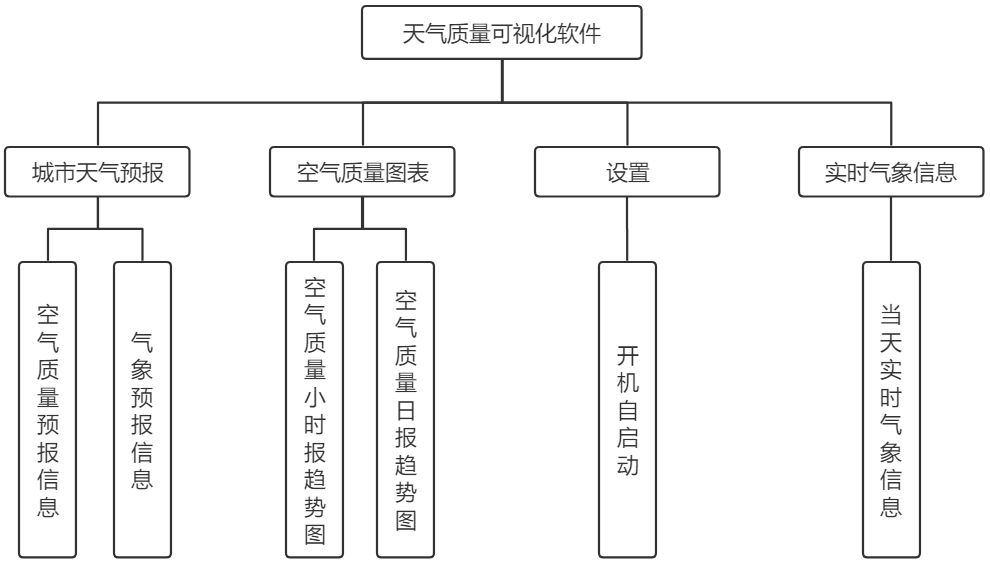


图4-1 程序功能结构

## 4.4 系统功能概述

1. 主界面展示

首先，在Qt组件中写好,写入后台服务器提供的数据接口，并针对主界面，通过传入的参数请求到相应的数据，最后通过方法遍历出数据，绑定到各标签中。点击菜单栏的相关内容就能将各个主界面显示出来。

2. 查询信息功能

在查询功能，实现只输入城市名或者点击城市按钮就可以一次性查询到气象预报、空气质量预报、实时气象信息、空气质量小时报趋势图和空气质量日报趋势图，且可以在模块化视图和趋势图之间进行切换查看。

3. 数据获取

在Linux系统中使用Scrapy框架进行爬取，使用crontab命令实现脚本隔段时间运行，实现数据的实时更新。

## 4.5 数据库设计

本项目中所采用的数据库为MySQL，创建数据库名称为weather\_quality。数据库weather\_quality中包含4张表。数据库设计在整个软件项目开发中是必不可少的一部分。数据库本身属于信息存储、操作的工具，数据库的作用是对软件所有天气的信息进行管理，数据库设计也有一定的流程规范，根据具体的系统需求来进行设计分析。先讨论数据库具体的模型，根据系统的需求分析及功能模块，进行数据库的设计[22]，逐步完成概念。逻辑的设计，最后利用相应工具将其建设为一个可靠的物理实体。

1. 实体属性

该系统中包括的实体主要为城市。数据库设计中经常使用实体关系图分析其属性集及各实体之间的联系，设计了E-R模型(entity-relationship model)图，为系统关系数据库的设计奠定了基础，本文给出了系统设计的各个功能模块和结构框架,阐述了实现了相关功能的过程,设计思路和E-R图。下面将利用E-R图对部分实体属性的进行介绍：

（1）气象预报实体属性图，图中椭圆表示其所携带的包括ID、星期、时间、风力等在内的一系列属性。气象预报实体属性图如图4-2所示。

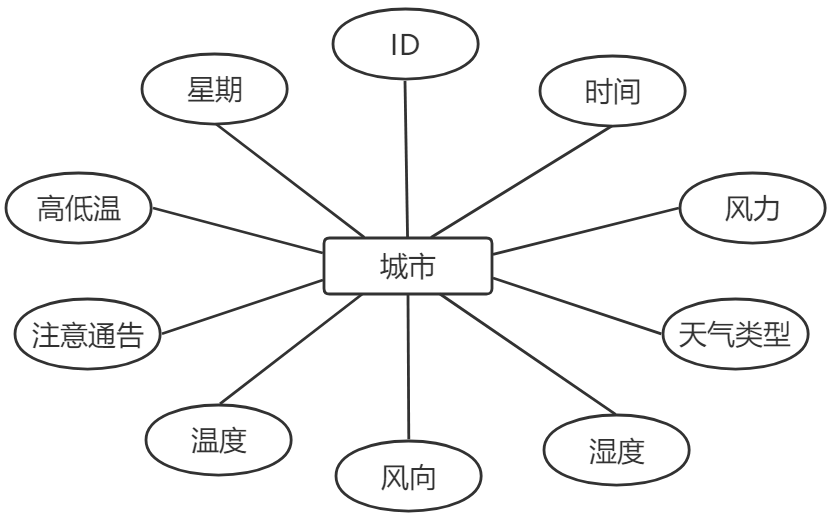


图4-2 气象预报实体属性图

（2）空气质量预报实体属性图，包括属性有近几天主要污染物、近几天天气质量指数、近几天天气质量。空气质量预报实体属性图如图4-3所示。

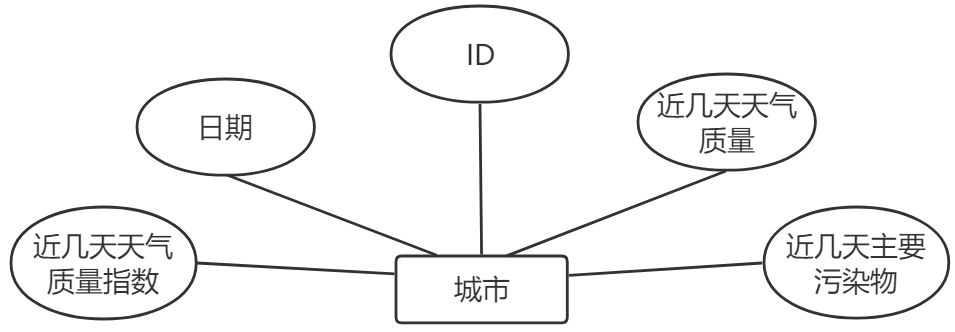


图4-3 空气质量预报实体属性图

（3）空气质量小时报实体属性图，日期时间、AQI、SO2、NO2等。空气质量小时报实体属性图如图4-4所示。

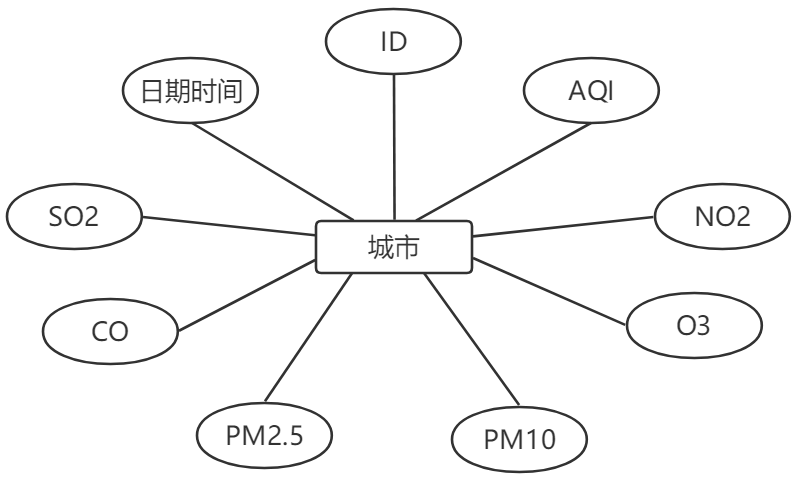


图4-4 空气质量小时报实体属性图

（4）空气质量日报实体属性图，日期、AQI、SO2、NO2等。空气质量日报实体属性图如图4-5所示。

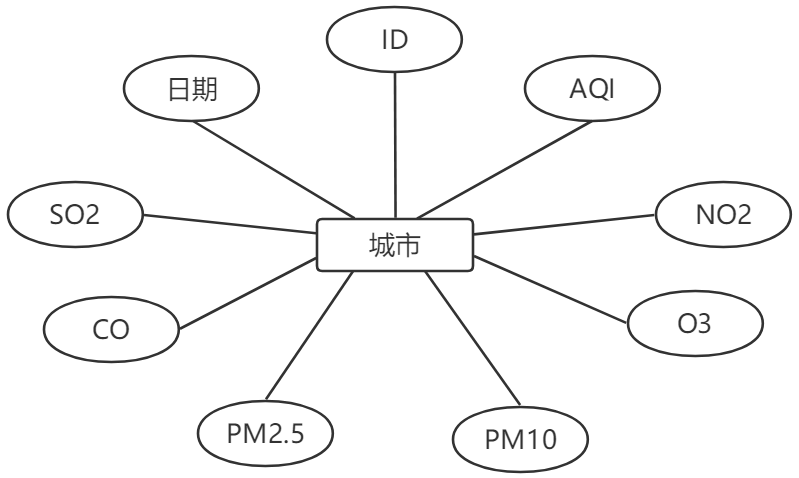


图4-5 空气质量日报实体属性图

2. 详细数据表说明

能创建数据库和数据表，并能对数据库中的数据进行增加、删除、修改和查询等操作，完成数据库的设计、管理和维护。本软件数据库中的数据表用于存放天气质量相关信息，以下是天气质量可视化软件各个数据库表进行说明：

气象预报表（temperaturedata）共包括10个字段，分别是id、cityname、times、week、high\_low、temperature、shidu、fx、fl、types、notice表示城市ID、城市名、时间、星期、高低温、温度、湿度、风向、风力、天气类型、注意通知。气象预报表如表4-1所示。

表4-1气象预报表 temperaturedata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 主键 | 说明 |
| id | char | 5 |  | ID |
| cityname | char | 10 |  | 城市名 |
| times | char | 20 |  | 时间 |
| week | char | 10 |  | 星期 |
| high\_low | char | 10 |  | 高低温 |
| temperature | char | 3 |  | 温度 |
| shidu | char | 5 |  | 湿度 |
| fx | char | 10 |  | 风向 |
| fl | char | 10 |  | 风力 |
| types | char | 20 |  | 天气类型 |
| notice | char | 20 |  | 注意通知 |

空气质量预报表（predictiondata）包括6个字段，分别是城市ID、城市名、主要污染物预报、空气质量指数预报等。空气质量预报表如表4-2所示。

表4-2 空气质量预报表 predictiondata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 主键 | 说明 |
| id | char | 10 |  | ID |
| cityname | char | 10 |  | 城市名 |
| timepoint | char | 15 |  | 日期 |
| pullutant | char | 80 |  | 近几天主要污染物 |
| api | char | 80 |  | 近几天空气质量指数 |
| quality | char | 50 |  | 近几天空气质量 |

空气质量小时报表（timesdata）包括10个字段，分别是城市ID、城市名、日期时间、AQI、SO2、NO2、CO、O3、PM2\_5、PM10。为了保证每过一段时间会对数据进行一次更新，避免数据冗余需设置主键。空气质量小时报表如表4-3所示。

表4-3 空气质量小时报表 timesdata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 主键 | 说明 |
| id | char | 10 | 是 | ID |
| cityname | char | 10 |  | 城市名 |
| timepoint | char | 20 |  | 日期时间 |
| AQI | char | 20 |  | AQI |
| SO2 | char | 20 |  | SO2 |
| NO2 | char | 20 |  | NO2 |
| CO | char | 20 |  | CO |
| O3 | char | 20 |  | O3 |
| PM2\_5 | char | 20 |  | PM2.5 |
| PM10 | char | 20 |  | PM10 |

空气质量日报表（daysdata）包括10个字段，分别是城市ID、城市名、时间、AQI、SO2、NO2、CO、O3、PM2\_5、PM10。同空气质量小时报表一样为了避免数据冗余，需设置主键。空气质量日报表如表4-4所示。

表4-4 空气质量日报表 daysdata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 主键 | 说明 |
| id | char | 10 | 是 | ID |
| cityname | char | 10 |  | 城市名 |

续表4-4 空气质量日报表 daysdata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| timepoint | char | 20 |  | 日期 |
| AQI | char | 20 |  | AQI |
| SO2 | char | 20 |  | SO2 |
| NO2 | char | 20 |  | NO2 |
| CO | char | 20 |  | CO |
| O3 | char | 20 |  | O3 |
| PM2\_5 | char | 20 |  | PM2.5 |
| PM10 | char | 20 |  | PM10 |

## 4.6 查询模块

查询模块实现了查询的功能，可以查询城市的气象预报信息、空气质量曲线图等相关信息。查询流程图如图4-6所示。

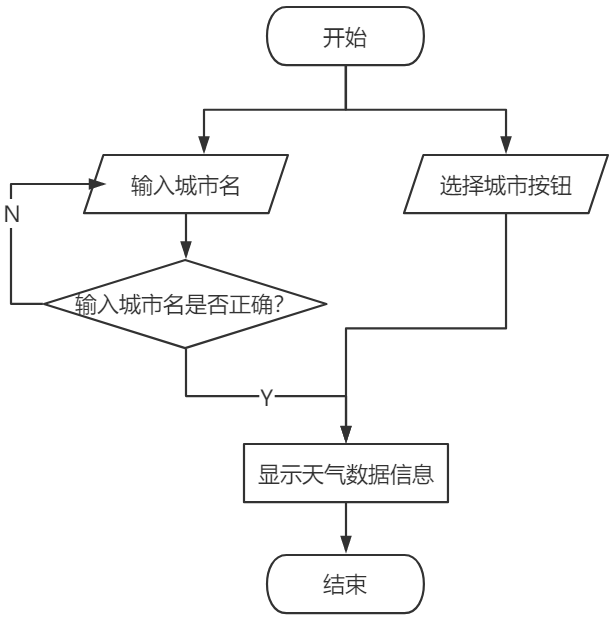


图4-6 查询流程图

当输入城市名验证是否正确，或者选择城市按钮后，软件各个模块更新天气质量数据。

## 4.7 本章小结

在本章初始部分，根据系统设计原理及目标提出了设计方案。之后根据设计方案进行了系统的总体架构，并对数据库进行了详细设计。最后对查询模块进行了详细设计。

# 第五章 系统实现

本章着重讨论可视化天气质量软件的实现过程，将设计和开发中遇到的问题总结为如何快速提取接口数据、服务器搭建、滚动列表实现、二维趋势图如何在Qt控件上交互四个方面，并对个四个方面进行详细的阐述。

## 5.1 气象和空气质量信息提取

在进行天气质量数据可视化之前，要对所需数据进行获取，获取天气数据的方式有很多种，可以通过中国气象数据网公布的API、气象数据统一服务接口CIMISS，还可以通过一些企业提供的API来进行获取等等。本论文通过四川生态环境厅的API获取城镇预报数据，使用中国天气API作为备用获取方式。二者数据取并集，互为备份。在获取到数据之后，对获取到的数据进行预处理，检查数据是否存在缺值或异常值，若存在缺值或异常值则通过备用API进行获取，保证数据的完整性和可用性[23]。

### 5.1.1 选择及分析数据

使用Fiddler抓包软件从两个网站上抓到了对应的信息接口，通过相应的接口配置参数（时间、站点、要素等），可以获取所需的天气预报。城市精细化预报产品要素资料接口如表5-1所示。

表5-1 城市精细化预报产品要素资料接口

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 接口ID | 接口名称 | 参数描述 |
| 1 | getAllCityDayAQIC | 按天检索精细化城市预报要素数据 | CITYNAME：城市名  TIMEPOINT：当天日期时间戳  AQI：空气质量指数  SO2：二氧化硫  NO2：二氧化氮  CO：一氧化碳  O3：臭氧  PM2.5：细颗粒物  PM10：可吸入颗粒物 |
| 2 | getAllCityRealTimeAQIC | 按时间检索精细化城市预报要素数据 | CITYNAME：城市名  TIMEPOINT：当前时间的时间戳  AQI：空气质量指数 |

续表5-1 城市精细化预报产品要素资料接口

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | getAllCityRealTimeAQIC | 按时间检索精细化城市预报要素数据 | SO2：二氧化硫  NO2：二氧化氮  CO：一氧化碳  O3：臭氧  PM2.5：细颗粒物  PM10：可吸入颗粒物 |
| 3 | getCityAuditResult | 按天检索往后几天城市预报要素数据 | CITYNAME：城市名  TIMEPOINT：当天日期时间戳  PULLUTANT：主要污染物  AQI：空气质量指数  QUALITY：空气质量评测 |

通过气象数据统一服务接口获取到所需的气象服务预报要素数据，首先需要对接口进行访问；然后，调用接口前要先看接口参数描述中的参数的赋值方式、参数是否为必选，以及接口对资料检索时间的限制条件等信息；最后，通过正则表达式，或者切片格式化数据并存储到数据库。

### 5.1.2 构造URL访问接口

拿空气质量预报信息为例，接口URL如下：

http://www.scnewair.cn:6112/smartadmin/forecast/getCityAuditResult?timePoint=1589198400000

经过分析，先使用Scrapy中的spider获取当天之后6天的空气质量信息需要将基础的URL加上当天日期的时间戳乘以1000构造成完整的空气质量信息URL，然后进行访问。

关键代码如下：

import time

urls=["http://www.scnewair.cn:6112/smartadmin/forecast/getCityAuditResult?timePoint="]

timeArray = time.strptime(now, "%Y-%m-%d") # 转换日期类型

timeStamp = int(time.mktime(timeArray)) \* 1000

url = self.urls[0] + str(timeStamp)

## 5.2 数据的存储

在进行天气数据可视化的过程中，首先时涉及天气数据的获取和存取，这就需要天气数据进行切片提取，下面对天气数据的切片过程进行介绍。

首先进行数据分析，拿成都市的小时报为例，接口数据如下：

{"data":[{"columns":{

"CITYCODE":5101,

"CITYNAME":"成都",

"AQI":"109",

"SO2":"5",

……

"TIMEPOINT":1589378400000}}],

"timePoint":1589378400000}

可以轻易观察到AQI、CO等数据以字典的形式存储在data键中，接下来需要对其进行提取，使用Scrapy框架中的piplines对数据进行处理提取，关键代码如下：

for i in range(len(datas)):

data = datas[i]["columns"]

id = str(time.localtime().tm\_mday) + "\_0\_" + str(i)

cityname = data["CITYNAME"]

timepoint = data["TIMEPOINT"]

AQI = data["AQI"]

SO2 = data["SO2"]

……

因为城市数据有很多，所以这里使用了for循环进行遍历提取。最后就是对数据进行存储，在此之前，需要连接MySQL数据库并创建数据库和对应的数据表。关键代码如下：

# 创建数据表

sql = "CREATE TABLE timesdata(" "id char(20)," "cityname char(20), " "timepoint char(20)," "CODE char(6)," "AQI char(20)," "SO2 char(20),……)"

cursor = db.cursor() # 获取游标对象

try:

cursor.execute(sql)

except Exception as info:

print(info)

# 插入数据

insert = "insert into timesdata(id,cityname,timepoint, AQI,SO2，……) values ('%s', '%s', '%s', '%s', '%s', ……)" % (id, cityname, timepoint, AQI, SO2,……)

cursor.execute(insert)

db.commit()

城市小时报数据如图5-1所示。

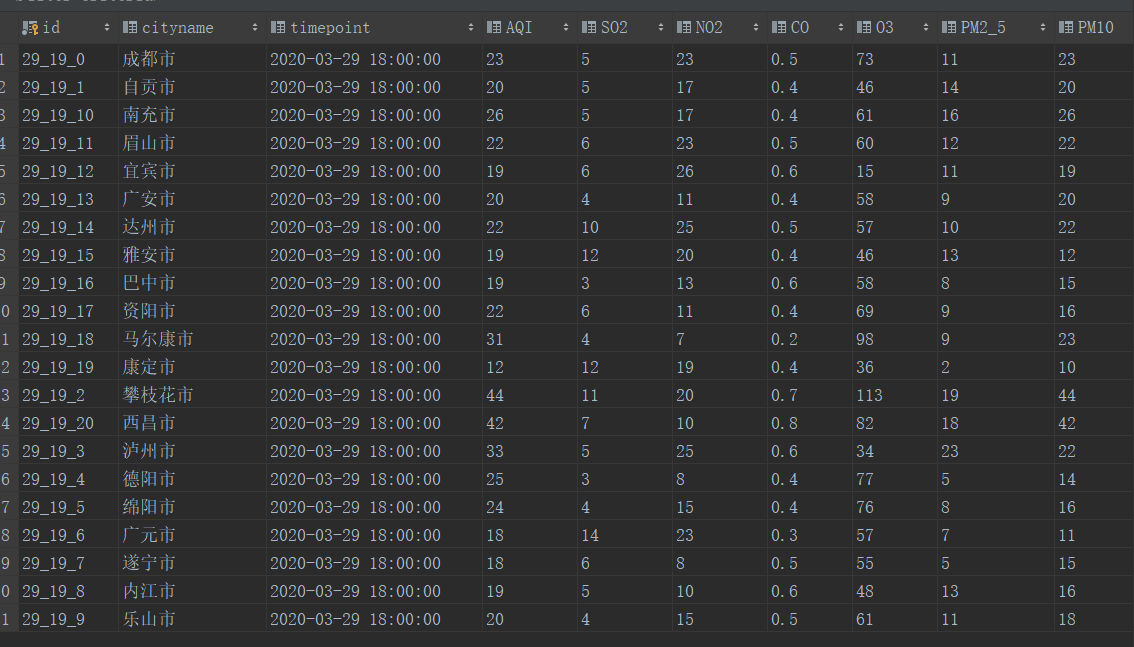


图5-1 城市小时报数据

## 5.3 Socket数据传输

TCP是面向连接的、可靠的基于字节流的传输层通信协议。面向连接的Socket通信流程如图5-2所示。

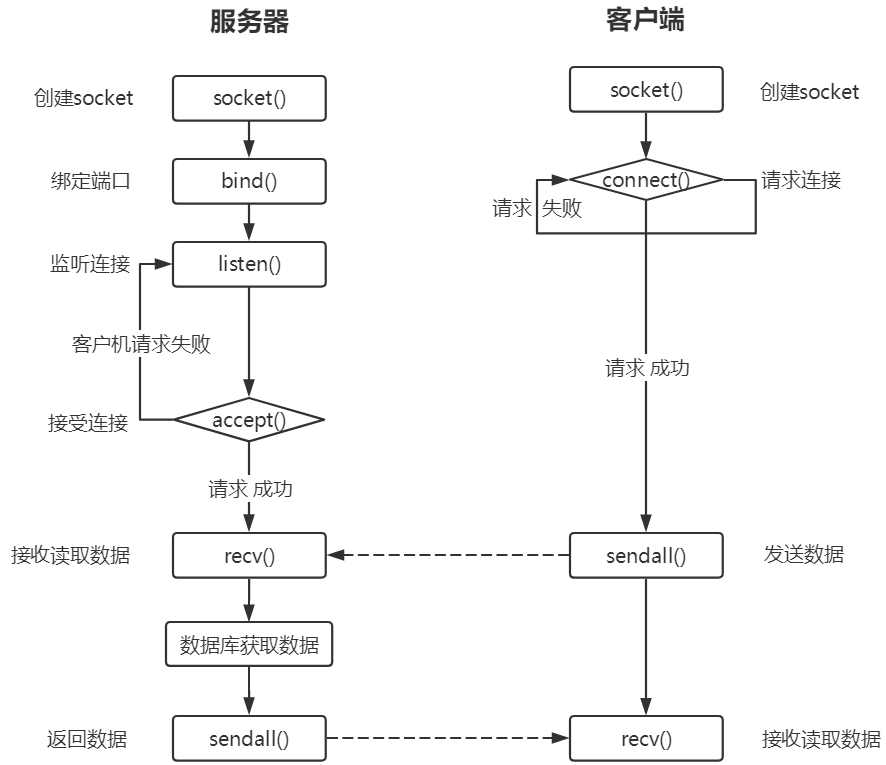


图5-2 面向连接的Socket通信流程

### 5.3.1 Client客户端

基于Python的交互式天气质量可视化软件采用前后端分离的开发模式，前端基于Pyqt5平台进行开发，只负责用户界面的展现，后端基于Python进行开发，负责可视化软件的业务逻辑。客户端与服务器交互采用Socket套接字响应式进行开发。同时，使用Pyecharts库作为客户端二维趋势图数据可视化的开发，使用js动态交互技术实现前后端数据交互。

### 5.3.2 Server服务器

基于Python的交互式天气质量可视化软件采用Linux环境下的PyCharm IDE作为服务器端开发平台，它是可以帮助用户在使用Python语言开发时提高其效率的工具，支持众多插件使得开发更加灵活。作为服务器，需要设计监听客户端请求的数量、接收数据的接口，反馈数据的接口，判断数据是否的功能，和错误处理机制。

## 5.4 可视化编程

可视化界面是使数据更好地与用户交互的最有效方式，所以页面的布局尤为重要，为了简便快速设计出天气质量可视化界面，首先使用Qt进行简单页面布局[24]。可视化界面UI设计如图5-3所示。



图5-3 可视化界面UI设计

导出成.ui文件转换为.py文件后就可以进行细节的布局了。

1. 菜单切换窗体功能的实现：

此功能需要使用到QlistWidget和QstackedWidget控件，这两个是实现选择左边菜单后，右边窗体切换到对应界面。

1. 可滚动式城市列表实现：

此功能是难点之一，在使用for循环生成城市列表后，每一个城市按钮需要进行特殊的信号槽设置，发送经过格式化的命令。

使用sender()函数用来记录发出信号的对象,如果单纯使用text()函数，返回的将会是列表中最有一个城市按钮的信息，使用sender()时主窗口class类的括号内必须是QMainWindow,不能是object。

1. 二维趋势图的生成与嵌套：

为了使用户观看天气质量信息数据更方便，二维趋势图是不二之选，所以在Python中有个将数据转换为js样式的二维趋势图的库Pyecharts，但是生成的文件又如何嵌套进Qt窗口中呢，于是使用到了QwebEngineView控件，可将网页嵌入到控件中。

## 5.5 软件主要功能

### 5.5.1 软件主要界面

根据上述章节的研究设计，用户打开天气质量可视化软件XM-Weather的图形用户界面的实现效果如下所示。

模块①：菜单列表，“城市空气预报”视图，以二维模块的方式展示天气质量信息，“空气质量图表” 将历史数据以二维趋势图的形式展示，“设置”该菜单可设置软件开机自启动。菜单列表如图5-4所示。



图5-4 菜单列表

模块②：城市搜索框，搜索数据库已经存在城市的天气质量信息，如果输入城市名错误或者数据库里没有该城市信息，则会弹出提示信息。城市搜索框如图5-5所示。

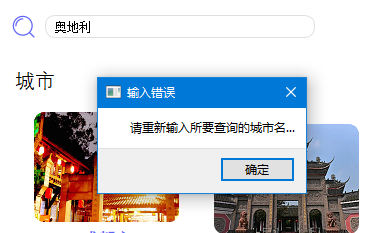


图5-5 城市搜索框

模块③：城市信息列表，当选中某城市或者搜索某城市，对应城市按钮则会上移突出显示，当鼠标悬浮在列表上时，城市按钮也有相应变化，默认会选中成都市。城市列表如图5-6所示。



图5-6 城市列表

模块④：城市空气质量预报，搜索某城市，该模块显示这城市近6天的空气质量预报。城市空气质量预报如图5-7所示。

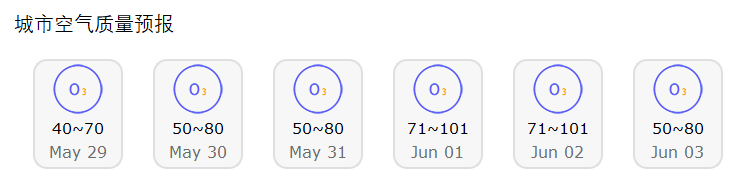


图5-7 城市空气质量预报

模块⑤，城市气象预报，搜索某城市，该模块显示这城市近15天的气象信息预报。城市气象预报如图5-8所示。



图5-8 城市气象预报

模块⑥：当天天气信息，显示选中城市当天的天气信息，包括温度、湿度、降水量、天气状态等信息。当天天气信息如图5-9所示。



图5-9 当天天气信息

模块⑦：预报类型，可以选择小时报或者日报，将会在模块⑧进行显示。日报趋势图如图5-10所示。

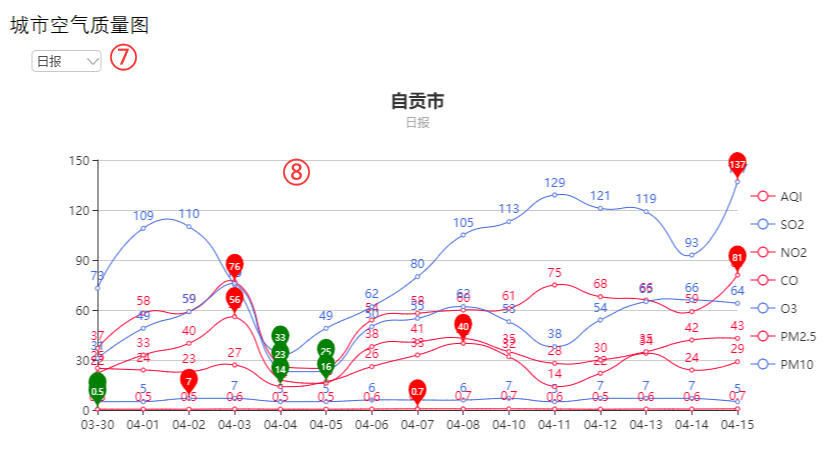


图5-10 日报趋势图

### 5.5.2 最小化至系统托盘

软件开启后软件可缩小到系统托盘，实现方便查看且不影响操作的功能，系统托盘图标如图5-11所示，关键代码如下。

tray = QtWidgets.QSystemTrayIcon(MainWindow) # 创建系统托盘对象

tray.setIcon(QtGui.QIcon('./icon/logo.png')) # 设置系统托盘图标



图5-11 系统托盘图标

### 5.5.3 服务器响应

当启动客户端后，客户端会向服务器发送连接信息，同时服务器会将默认城市的天气质量信息一并返回给客户端，服务器响应如图5-12所示，左边是服务器收到的命令，右边是客户端收到的部分数据的展示。

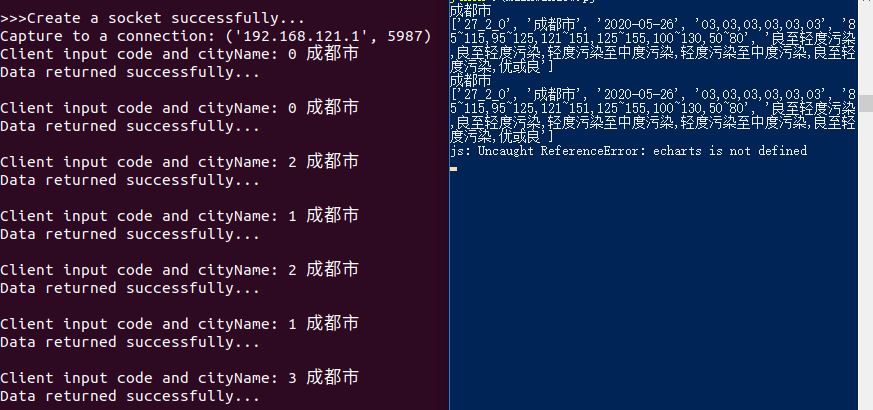


图5-12 服务器响应

### 5.5.4 自动化爬虫

为使软件更完整，自动化是最好的选择，所以自动化爬虫的实现使用到了Linux中的crontab命令，首先需要设置运行py文件的shell脚本文件getDaysdata.sh，关键代码如下：

#！/bin/bash

Export PATH=$PATH:/usr/local/bin

cd /home/walden/Server\_Weather/spiders && scrapy crawl days –nolog

然后使用crontab -e开启自动运行的脚本，将日志记录到spider.log文件中，关键代码如下：

15 8 \* \* \* bash /home/walden/Server\_Weather/spiders/getDaysdata.sh >> /home/walden/Logs/spider.log

## 5.6 小结

本章介绍了基于Python的可视化天气质量软件的实现，对软件的设计目标、总体框架、总体设计进行了详细的描述，并对该软件的主要功能进行了展示。

# 第六章 系统测试

## 6.1 测试的目的与目标

在此系统进行初步实现之后，开始进行对系统进行测试，找出系统中存在的Bug，通过测试，用提交的Bug报告来为以后软件的改进提供标准和参考，能够在以后的系统改进中找到依据。

测试后的软件各模块基本功能可以顺利进行，尽可能的提高软件的健壮性。

## 6.2 测试方法

• 从是否关心软件内部结构和具体实现的角度划分：黑盒测试和白盒测试；

• 从是否执行程序的角度：静态测试和动态测试；

• 从软件开发的过程按阶段划分有：单元测试、集成测试、确认测试、系统测试、验收测试、回归测试、Alpha测试、Beta测试；

单元测试又称模块测试，是针对软件设计的最小单位 ─ 程序模块（这里所说的程序模块在Java中一个模块就是一个方法），进行正确性检验的测试工作。其目的在于发现各模块内部可能存在的各种差错。单元测试需要从程序的内部结构出发设计测试用例。多个模块可以平行地独立进行单元测试。

集成测试 (组装测试、联合测试），通常在单元测试的基础上，需要将所有模块按照设计要求组装成为系统。这时需要考虑的问题是：

• 在把各个模块连接起来的时候，穿越模块接口的数据是否会丢失；

• 一个模块的功能是否会对另一个模块的功能产生不利的影响；

• 各个子功能组合起来，能否达到预期要求的父功能；

• 全局数据结构是否有问题；

• 单个模块的误差累积起来，是否会放大，从而达到不能接受的程度。

确认测试（Validation Testing），确认测试又称有效性测试。任务是验证软件的功能和性能及其它特性是否与用户的要求一致。对软件的功能和性能要求在软件需求规格说明书中已经明确规定。它包含的信息就是软件确认测试的基础。

系统测试（System Testing），是将通过确认测试的软件，作为整个基于计算机系统的一个元素，与计算机硬件、外设、某些支持软件、数据和人员等其它系统元素结合在一起，在实际运行环境下，对计算机系统进行一系列的组装测试和确认测试。系统测试的目的在于通过与系统的需求定义作比较， 发现软件与系统的定义不符合或与之矛盾的地方。

验收测试（Acceptance Testing），在通过了系统的有效性测试及软件配置审查之后，就应开始系统的验收测试。验收测试是以用户为主的测试。软件开发人员和QA（质量保证）人员也应参加。由用户参加设计测试用例，使用生产中的实际数据进行测试。在测试过程中，除了考虑软件的功能和性能外，还应对软件的可移植性、兼容性、可维护性、错误的恢复功能等进行确认。

## 6.3 测试用例

由于软件功能模块分类较多，测试内容篇幅较长，所以在本论文中只介绍从搜索栏发送数据和数据库返回对应数据的测试用例，发送和返回数据测试用例如表6-1所示，从测试的结果来看与期望结果完全相同。

表6-1 发送和返回数据测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能特性 | 发送命令获取数据 | | | | |
| 测试目的 | 验证服务器是否接收命令，做判断后并返回数据 | | | | |
| 测试数据 | 城市名：成都、成都市、test | | | | |
| 测试内容 | 操作描述 | 数据 | 期望结果 | 实际结果 | 测试状态 |
| 1 | 在搜索栏输入城市名不加“市” | 成都 | 服务器接收数据，进行判断，并在数据库搜索对应城市天气质量数据，最后返回数据 | 服务器接收数据，进行判断，成功从数据库搜索出相应城市数据 | P |
| 2 | 在搜索栏输入城市名 | 成都 | 服务器接收数据，进行判断，并在数据库搜索对应城市天气质量数据，最后返回数据 | 服务器接收数据，进行判断，成功从数据库搜索出相应城市数据 | P |
| 3 | 在搜索栏输入英文 | test | 服务器接收数据，进行判断，发现数据库无此城市名，返回错误命令 | 服务器接收数据，进行判断，发现数据库无此城市名，返回错误命令 | P |

## 6.4 测试结论

把开始的代码写得越好，它出现的错误也就越少，你也就越能相信所做过的测试是彻底的。系统化测试以一种有序方式设法探测潜在的麻烦位置。同样，毛病最可能出现在边界，这可以通过手工的或者程序的方式检查。自动进行测试是最理想的，用得越多越好，因为机器不会犯错误、不会疲劳、不会用臆想某此实际无法工作的东西能行来欺骗自己。回归测试检查一个程序是否能产生与它们过去相同的输出。在做了小改变之后就测试是一种好技术，　能帮助我们将出现问题的范围局部化，因为新问题一般就出现在新代码里面。

# 第七章 结论与展望

## 7.1 结论

在大力发展互联网+的时代，各个领域在气象数据方面的应用层出不穷，天 气预报作为与人们息息相关的气象服务，它的宗旨就是让人们更直观、更全面的掌握天气情况。本论文作为空气、气象观测数据二维可视化系统的子系统之一，构建了一个操作简单、界面友好、展示直观的天气预报三维可视化系统，能帮助用户直观、快速的了解天气，从而有效的进行防范，具有很强的实用性。主要工作总结如下：

（1） 分析了国内外数据可视化技术和基于Python的可视化天气质量软件的研究，对当前的天气预报产品进行了仔细地对比分析，发现了它们的共同点，找出了它们存在的问题，提出了研究的主要方向。

（2） 对课题研究所涉及的有关数据可视化、Scrapy框架、Socket、PyQt5等技术进行了深入的学习和研究，给出了相应关键技术的实现。

（3） 对软件开发所要解决的关键问题，地图的设计、发布、接入、显示，天气预报数据的二维可视化技术，js动态交互技术都进行了深入的研究。

（4） 设计与实现了软件的数据访问层、业务逻辑层和表示层三层体系结构， 明确了每一层的内容和职责。

（5） 实现了天气质量预报数据的二维可视化软件。对软件的设计目标、处理框架、总体设计、软件服务器端搭建进行了详细的设计和实现。本论文研发的基于Python的可视化天气质量软件能有效地解决当前天气预报产品存在的展现形式单一、缺乏交互等问题，使公众能够更高效、更直观的掌握天气情况。

## 7.2 展望

虽然本论文提出了在天气预报二维可视化方面具有创造性的研究方向，并完成了一定的效果，但由于研究条件和时间方面的有限，数据展示没有达到最理想的效果，需要进行进一步的改进和完善，下一步工作包括以下几个方面：

（1） 扩大对天气质量数据展示的覆盖面。目前对天气数据的展示只有天气状况、降水量、气温、空气质量AQI这四个要素，只能展示未来6天，下一步要逐步增加对各类天气数据的展示，能够查询未来更远的天气情况，并且能够查询到更多城市的天气质量信息。

（2） 实现天气预报个性展示。目前只是对用户的操作数据进行了收集，下一步应该研究如何利用这些操作数据为用户提供个性化的天气预报服务。

（3） 减少图标加载的速度。由于天气数据的实时性和大规模性，生成js动态信息文件需要一些时间，应该进一步研究如何高效地显示天气数据，增强用户体验。

致　谢

学生时光从我的指尖悄悄掠过，转眼间在四川轻化工大学的求学生活就要结束了，即将告别学生时代去迎接新的挑战，四年时间我成长了许多，也学会了许多，在这里我要对在我学习和生活上给予帮助的人表示由衷的感谢。

首先，我要感谢我的导师石睿老师。石老师平易近人，对待工作一丝不苟，事无巨细，不仅给我们传授书本上的知识，而且教会了我许多做人的道理，我还要感谢我们的辅导员杨俊老师和蒋欢老师，本科四年给了我很大的帮助与关心，其次，我要感谢计科167班的伙伴，四年时间我们一起生活一起成长，每当我遇到困难，他们总向我伸出援助之手，为我解惑，给我帮助，使我获得了很多学习经验，他们优秀的品质让我向他们看齐。最后，我要感谢我的家人，在我多年的学习生活上义无反顾地支持我、鼓励我、关心我，当别人关注的是我飞的高不高的时候，是他们关心的是我飞的累不累，家，就是我的避风港。

流光容易把人抛，红了樱桃绿了芭蕉。每一个出现在我们生命中的人，都是注定的缘分，在今后的十年、二十年、三十年……直到永远，不论我们身在何处，我都会把你们放在心中，永远的铭记你们。

参考文献

[1] 白珍. 气象数据图形显示系统的设计与实现[D]. 四川: 电子科技大学, 2018.

[2] 何雪莹. 探索性数据可视化分析系统的设计与实现[D]. 四川: 西南交通大学, 2017.

[3] 李江. 复杂空间数据三维可视化关键技术研究[J]. 科技创业月刊, 2017, (9):197-198.

[4] 赵婷, 华一新, 李响,等. 一种基于HeatMap的地理标签数据可视化表达的研究[J]. 测绘工程, 2018, 25(6):28-32.

[5] 李万锋. 基于Java EE的数据可视化平台设计及主要功能实现[D]. 北京: 中国科学院大学(中国科学院工程管理与信息技术学院), 2017.

[6] 金岩. 基于GIS的数据展示平台的设计与实现[D]. 山东: 山东大学, 2018.

[7] 严丙辉. 结合地理信息的气象数据可视化平台设计与实现[D]. 浙江: 浙江大学, 2018.

[8] 陈俊生, 彭莉芬. 基于Python+Echarts的大数据可视化系统的设计与实现[J].安徽电子信息职业技术学院学报, 2019, 18(4):6-9+22.

[9] 肖宇. Python在数据库教学中的应用[J]. 电脑知识与技术, 2020, 16(6): 152-153.

[10] 蔡敏. Python语言的Web开发应用分析[J]. 无线互联科技, 2019, 16(4): 27-28.

[11] （美）迪米特里奥斯·考奇斯·劳卡斯．精通Python爬虫框架Scrapy[M]. 北京:人民邮电出版社，2018.

[12] 胡松涛．Python网络爬虫实战[M]. 北京: 清华大学出版社，2017.

[13] 郭锋锋. 基于python的网络爬虫研究[J]. 佳木斯大学学报(自然科学版), 2020, 38(2):62-65.

[14] Fontaine Rafamantanantsoa,Maherindefo Laha. Analysis and Neural Networks Modeling of Web Server Performances Using MySQL and PostgreSQL[J]. Communications and Network, 2018, 10(4): 63-72.

[15] 郑智方, 李彬, 刘世坤, 等. 探究mysql的运用实例-对数据库的宏观把握[J]. 科技风, 2020, (6): 129.

[16] 陶文玲, 侯冬青. PyQt5与Qt设计师在GUI开发中的应用[J].湖南邮电职业技术学院学报, 2020, 19(1):19-21.

[17] 侯文放, 王辉, 曾波. 数据中心网络中基于传输速率分配的TCP协议[J]. 计算机测量与控制, 2020, 28(3):227-230.

[18] 何振宇. 基于流量分析的HTTP协议安全现状分析[J]. 科学技术创新, 2020, (3):82-84.

[19] 田春婷. 基于Java Socket文字聊天系统的GUI实现[J]. 兰州石化职业技术学院学报, 2019, 19(2):11-15.

[20] 任延璞. 浅析软件工程需求分析与结构建设[J]. 无线互联科技, 2016, (22): 50-52.

[21] 刘佳, 王丽, 杨俊叶. 计算机软件测试方法及应用分析[J]. 科技风, 2020, (9): 119.

[22] 宋永生, 黄蓉美, 王军. 基于Python的数据分析与可视化平台研究[J]. 现代信息科技, 2019, 3(21):7-9.

[23] 刘喆玥. WebGIS技术在交互式天气预报三维可视化系统中的应用[D]. 南昌:南昌大学, 2018.

[24] 刘海燕, 马振涛. 构建Windows环境下Python GUI网络编程实验环境[J]. 北华航天工业学院学报, 2018, 28(4):4-6.

附录

## 附录A:软件使用说明书

1 软件概述

本设计要实现的是对气象信息和空气质量信息的可视化，在设计该软件时，应贴近生会，便于用户和研究人员对天气信息进行查看与观测，便于用户操作，主要是填补市面上天气质量信息可视化软件的空缺，能够实现数据查看的快速化，便捷化。本系统主要包括气象预报、空气质量预报、数据趋向图等模块，每一个模块中，都必须做到良好的输入及输出界面，能够实现针对某一种数据进行准确的观测的功能。

1.1 功能

系统在实现上应该如下功能:

1. 软件要求用户必须在联网的情况下进行查看数据。

2. 系统提供查询功能，便于用户对气象预报、空气质量预报、空气质量趋向图等相关信息查询。

3. 系统提供用户登录能够实现所有用户登录，不用任何操作自动显示当地天气质量的功能。

2 软件安装

2.1 系统要求

系统的安全是保证系统正常使用的首要保障，系统要从访问控制及数据完整方面考虑，对服务器系统的运行维护、操作方法都要有明确的说明。

2.2 安装前的准备

硬件准备：CPU Pentium Ⅳ 1.8Ghz，内存1GB，硬盘空间50.0MB以上硬盘剩余空间。

软件环境准备：搭建好服务器开发环境Scrapy、MySQL。和前端运行环境PyQt5。

2.3 安装

（1） 服务器环境安装（Linux系统）

第一步，安装PyCharm IDE，在官网下载安装Linux版本

第二步，安装python 3.7，在官网下载安装Linux版本

第三步，升级pip，在命令符界面输入

python -m pip install --upgrade pip

第四步，安装scrapy

pip install scrapy

第五步，安装mysql数据库

pip install pymysql

（2） 前端环境安装（Windows10系统）

第一步，安装PyCharm IDE，官网下载安装Windows版本

第二步，安装python 3.7，在官网下载安装Windows版本

第三步，安装PyQt5，在命令符界面输入

pip install pyqt5

pip install PyQt5-tools

第四步，安装Pyecharts，在命令符界面输入

pip install pyecharts

pip install pyecharts\_snapshot

pip install echarts-countries-pypkg

pip install echarts-china-provinces-pypkg

pip install echarts-china-cities-pypkg

3 运行说明

3.1 软件启动

检查电脑联网否，然后双击XM-Weather.exe可执行程序，就可以查看天气质量数据。

3.2 用户登录

用户可以选择是否登录账号，当然如果没有登录也可以正常使用程序，登录后会显示用户之前保存过的操作习惯，和显示所在地的城市天气质量信息。

## 附录B:主要源程序

### 单击左边菜单，右边切换到对应界面的功能实现。

#创建右边窗体

self.Right\_Pages = QtWidgets.QStackedWidget(self.Right\_Widget)

# 左边菜单

def setItemsPages(self):

self.List\_Widget.setFrameShape(QtWidgets.QListWidget.NoFrame)

list\_str = ['Dashboard', 'Statistics', 'Setting']

self.page = []

self.button = []

# 生成菜单

for i in range(len(list\_str)):

self.page.append(i)

self.button.append(i)

item = QtWidgets.QListWidgetItem(self.List\_Widget)

item.setText(list\_str[i])

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.List\_Widget.addItem(item)

# 右侧的窗体与对应菜单链接

self.page[i] = QtWidgets.QWidget(self.Right\_Pages)

self.page[i].setObjectName(f"page{[i]}")

self.Right\_Pages.addWidget(self.page[i])

# 列表与右侧的QWidgetIndex对应绑定

self.List\_Widget.itemClicked.connect(self.switch\_stack)

# 默认选中列表中的第一个

self.List\_Widget.setCurrentRow(0)

# 菜单按下跳转到第i个窗体

def switch\_stack(self):

try:

i = self.List\_Widget.currentIndex().row()

self.Right\_Pages.setCurrentIndex(i)

except:

pass

### 可滚动式城市列表实现。

for i in range(len(city\_name)):

self.City\_Button.append(i)

self.City\_Button[i] = QtWidgets.QPushButton(

self.scrollAreaWidgetContents)

self.City\_Button[i].setObjectName("City\_Button1")

self.City\_Button[i].setText(f"{city\_name[i]}")

# 此字典，存放按钮拼音名与其号数

self.City\_Dict[city\_pinyin[i]] = i

self.City\_Button[i].clicked.connect(lambda:bCustom.moveButton(self.sender().text()))

### 二维趋势图的实现。

time\_line = Line(init\_opts=opts.InitOpts(width="800px", height="300px"))

time\_line.add\_xaxis(timepoint[1]) # 增加x轴

time\_line.add\_yaxis("AQI", aqi[1],

is\_smooth=True, # 线条平滑

# 曲线填充面积的透明度

areastyle\_opts=opts.AreaStyleOpts(opacity=0),

color="#4169E1" )

……

day\_grid.render("./web/day\_line.html")

# 嵌入网页控件

self.browser = QWebEngineView()

self.browser.setHtml"./web/day\_line.html")

# 将网页嵌入到Vb\_Layout布局中

self.Vb\_Layout.addWidget(self.browser)

### 客户端

def \_\_init\_\_(self):

ADDR = ( '192.168.1.101', 9999)

self.BUFSIZ = 4096

# 创建套接字

self.Clt = socket.socket()

# 请求连接

self.Clt.connect(ADDR)

def sendData(self, city\_name):

while True:

# 给服务器发送信息

self.Clt.sendall(city\_name.encode())

# 接收服务器返回的信息

data = self.Clt.recv(self.BUFSIZ) # bytes类型

return json.loads(data.decode())

self.Clt.close()

### 服务器

while not isClose:

print("Create a socket successfully...")

Ser = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# socket默认不支持地址复用，但可以在绑定前使用setsockop函数让套接字允许地址重用

Ser.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

# 绑定端口

Ser.bind(ADDR)

Ser.listen(5)

# 接受连接

clnt, addr = Ser.accept()

while True:

try:

clt\_data = clnt.recv(BUFSIZ)

# 选择函数

t = clt\_data.decode().split(" ")[0]

city\_name = clt\_data.decode().split(" ")[1]

if not clt\_data:

print("Client did not send data... ")

except Exception as err:

print(err)

break

if t == '0':

default = SearchSQL().city\_name

……

clnt.sendall(json.dumps(default).encode())

Ser.close()