

《创新训练课B》结题报告

学院： 基础学部

题目： **模拟地铁自动售票系统**

班级：**工科实验班（机器人与智能装备）1班**

姓名： **姜储林**

学号： **2024312144**

教师：

日期：

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 25.7.3 | V1.0 | 1. 实现欢迎界面的UI功能 2. 实现按钮的信号槽机制 | 姜储林 |
| 25.7.15 | V1.1 | 1. 基于QT6与C++搭建了底层代码 2. 完成主菜单界面的UI功能 | 姜储林 |
| 25.8.1 | V1.2 | 1. 修复语法错误与BUG 2. 完成地图界面的UI设计 | 姜储林 |
| 25.8.15 | V1.3 | 1. 搭建完成地图界面的初始化、容器等代码 2. 初步实现缩放功能 | 姜储林 |
| 25.8.27 | V1.4 | 1. 修复已有BUG 2. 实现地图随指针缩放功能和鼠标按键平移 | 姜储林 |
| 25.9.16 | V1.5 | 1. 进行UI界面的统一与美化 2. 进行代码的模块化维护，增强可读性 | 姜储林 |
| 25.10.5 | V2.0 | 1. 设计售票界面UI 2. 整理地铁线路站点信息库 | 姜储林 |
| 25.10.15 | V2.1 | 1. 实现基于站点筛选的初始站和终点站选择 2. 实现基于统一线路站点判断的价格判断 | 姜储林 |
| 25.10.21 | V2.2 | 1. 实现基于线路、站点的双条件筛选 2. 实现找零功能 | 姜储林 |
| 25.11.4 | V2.3 | 1. 搭建线路图，实现基于BFS的距离判断 2. 改进价格判断，实现基于距离的价格机制 | 姜储林 |
| 25.11.7 | V2.4 | 1. 修复已有BUG 2. 对售票界面进行统一标准的美化 | 姜储林 |
| 25.11.8 | V3.0 | 1. 进行细节修复 2. 撰写用户说明手册 3. 打包跨平台测试 | 姜储林 |

**目 录**

[1 引言 1](#_Toc202188405)

[1.1编写目的 1](#_Toc202188406)

[1.2背景 1](#_Toc202188407)

[1.3定义 1](#_Toc202188408)

[1.4参考资料 1](#_Toc202188409)

[2 任务概述 1](#_Toc202188410)

[3 需求分析 1](#_Toc202188411)

[3.1 用户需求分析 1](#_Toc202188412)

[3.2 运行环境 1](#_Toc202188413)

[4 功能及操作介绍 1](#_Toc202188414)

[4.1 操作 1](#_Toc202188415)

[4.2 功能 1](#_Toc202188416)

[5 系统设计 1](#_Toc202188417)

[5.1 总体架构设计 1](#_Toc202188418)

[5.2 模块分析与设计 2](#_Toc202188419)

[5.3软件结构（流程图） 2](#_Toc202188420)

[6 调试与测试 2](#_Toc202188421)

[6.1调试过程 2](#_Toc202188422)

[6.2测试结果 2](#_Toc202188423)

[7 编程中遇到的问题 2](#_Toc202188424)

[7.1 问题1 2](#_Toc202188425)

[7.2 问题2 2](#_Toc202188426)

[8分析总结与心得体会 2](#_Toc202188427)

# 1 引言

## 1.1编写目的

基于QT框架设计跨平台地铁售票系统，实现购票全流程可视化，掌握QT信号槽机制、QGraphicsView图形渲染及多线程支付处理等关键技术。

## 1.2背景

地铁自助售票系统需高可靠性图形界面，QT的跨平台特性和丰富UI组件库可满足专业级应用需求。本项目为哈尔滨工业大学(深圳)机器人与智能装备专业 2025 年暑季小学期创新训练课 B 的实践作业。

## 1.3定义

Mainwindow类：主页面，用作菜单

Welcome类：欢迎界面，附带作者信息

Mapwindow类：地图，用于深圳市地铁信息的查询

Sale类：在线售票窗口，具有选择线路、站点等信息

## 1.4参考资料

《C++ primer（第5版）》Stanley B. Lippman 等合著。

《Qt 6 C++开发指南》王维波等合著.

官方文档：doc.qt.io

深圳地铁售票模拟方框图（课程设计题目2025.pdf）

# 2 任务概述

系统的功能需求描述如下：

（1）显示欢迎界面，作者信息和版权信息。

（2）进入系统主菜单，提供购票选项、地图查询选项、退出系统，三个选项。

（3）系统说明界面详细的介绍了购票流程，并且附有用户须知。

（4）用户选择开始购票，进入始发站选择界面，或者由此返回主界面。

（5）用户选择好了始发站后进入终点站的选择，或者由此返回主界面。

（6）根据系统提示然后进入票数的选择，或者由此返回主界面。

（7）根据系统提示进入投币找币流程，或者由此返回主界面。

（8）购票成功。

# 3 需求分析

## 3.1 用户需求分析

用户在深圳市地铁进行售票操作时，会有查询地图、搜寻地铁线路与站点、选择人数、支付地铁票找零等。本项目旨在实现和改良真实的售票系统，美化售票界面与规范化管理维护代码。

## 3.2 运行环境

操作系统：Windows 11

操作平台：QT Creator 17.0.0

使用语言：“QT6”+“C++”

# 4 功能及操作介绍

## 4.1 操作

点击打包好的exe执行文件，进入欢迎界面。



图1 欢迎界面

点击Next Page按钮，进入主菜单，主菜单中包含具体的操作流程和温馨提示，主菜单有三个按钮，分别是购票、地图查询和退出。地图查询进入后可以查询深圳市地铁线路与站点详细信息，而后进入购票处，进行站点、线路双信息筛选购票。

图2 主菜单界面

进入地图查询界面，可以进行基于光标的缩放和拖动平移，便于地图的查询。

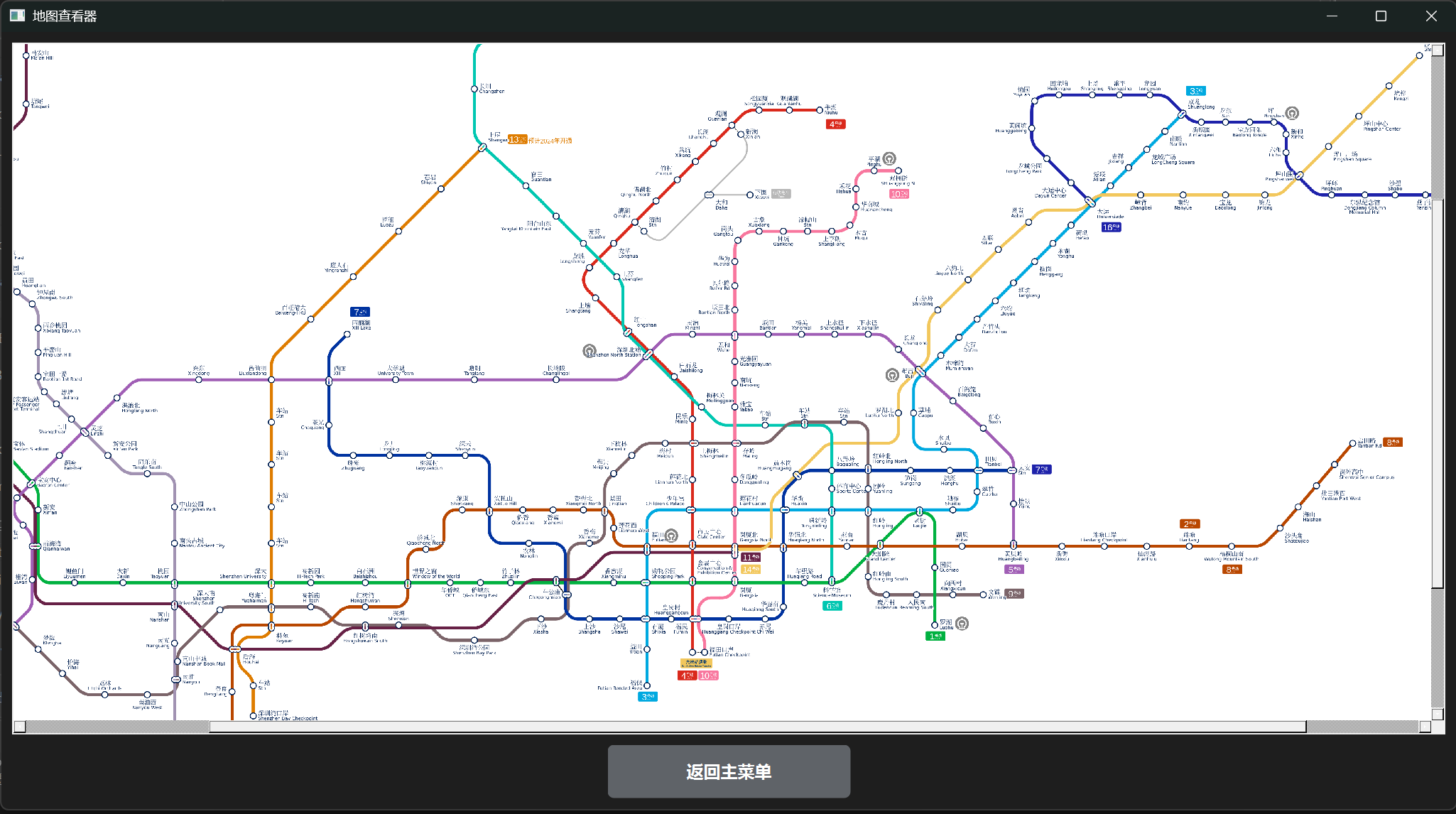


图3 地图查询界面

进入购票界面，可以根据线路、站点进行对应起始站和终点站的下拉筛选，同时选择站点后会有提示信息该站点所在的线路，提示信息。在选好站点后进行人数选择，可以进行鼠标滚轮或者对话框右边按钮进行调整，票价会根据人数和单票进行实时反馈，在输入完金额确认出票后，会出现两种情况，第一种为输入金额不足以买票，会弹窗提示还差的金额数量；第二种为输入金额足以购买地铁票，会弹窗出票成功。（此处限制输入金额为非负整数）



图4 售票界面展示



图5 线路下拉筛选展示



图6 站点下拉筛选展示



图7 支付金额不足以票价展示



图8 支付金额足以支付票价展示

在售票和地图查询界面点击返回主菜单按钮即可返回主菜单，方便随时进行更新，支持多次票价的计算，同时地图的反复查询。

## 4.2 功能

1）用户界面系统：

欢迎界面（作者信息展示）

主菜单界面（售票、地图查询、退出）

售票操作界面（线路选择、站点选择、人数选择）

地图查询界面（深圳市地铁线路可视化）

1. 核心业务功能：

地铁票价计算（基于深圳地铁官方计费规则）

支付与找零系统

地图交互功能（缩放、平移）

1. 辅助功能：

帮助文档系统

界面美化与用户体验优化

# 5 系统设计

## 5.1 总体架构设计

本系统采用模块化设计理念，基于QT6框架构建，整体架构分为三层：

用户界面层：

欢迎界面（Welcome）

主菜单界面（MainWindow）

地图查询界面（MapWindow）

售票界面（Sale）

底层逻辑层：

地图交互控制模块

票价计算引擎

支付处理系统

数据管理模块

数据层：

深圳地铁线路数据库（25条线路）

站点坐标信息库

票价规则数据库

## 5.2 模块分析与设计

1）地图查询模块

功能：实现深圳市地铁线路可视化展示

类：Mapwindow

关键技术：基于QT6自带的函数库 <QGraphicsView>、<QGraphicsScene>、<QGraphicsPixmapItem>与<QWheelEvent>实现光标缩放与拖动平移。

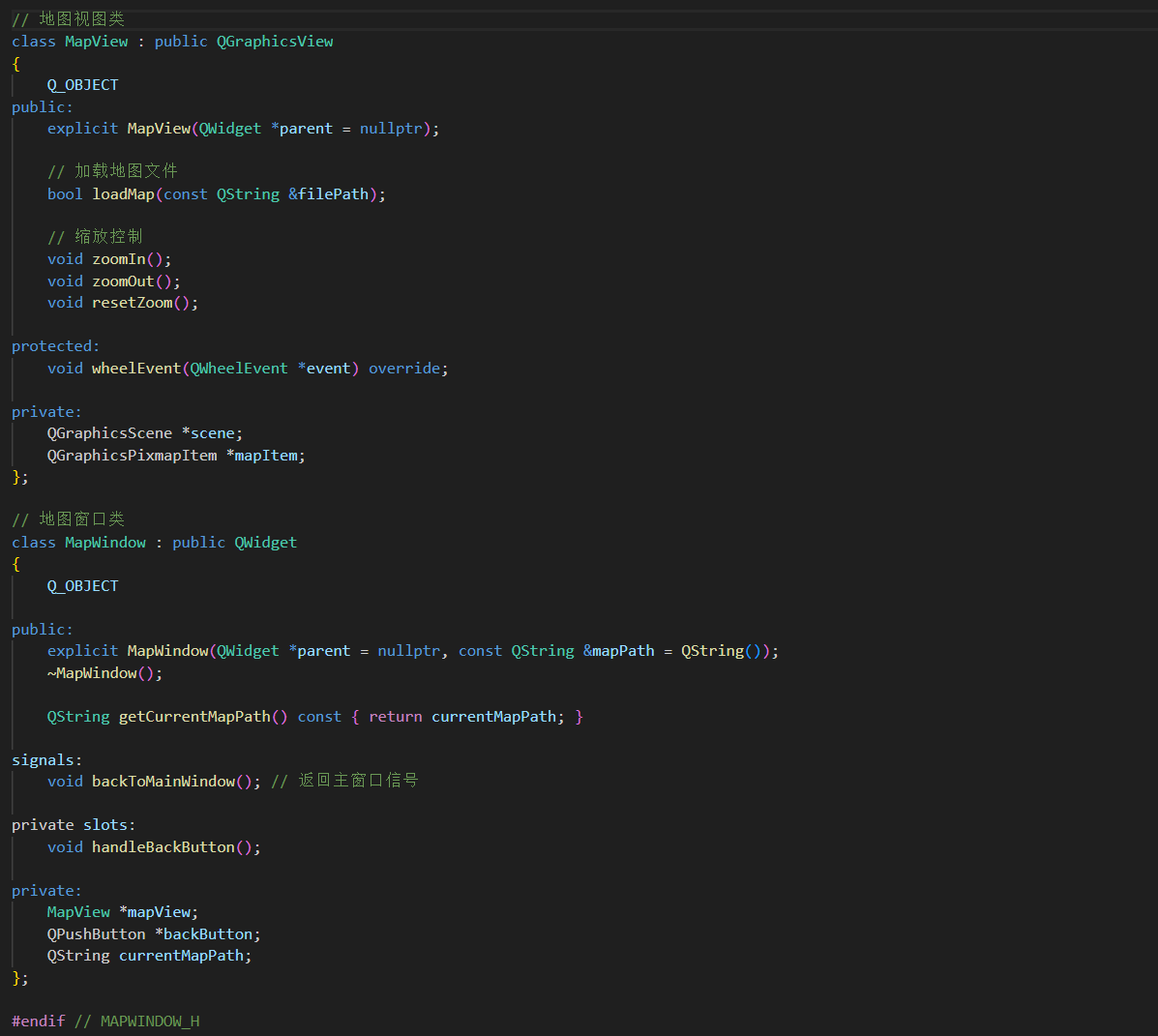


图9 Mapwindow.h主要函数

Mapwindow.cpp核心功能：

1. 初始化代码（部分）



图10 地图初始化与加载代码（part1）

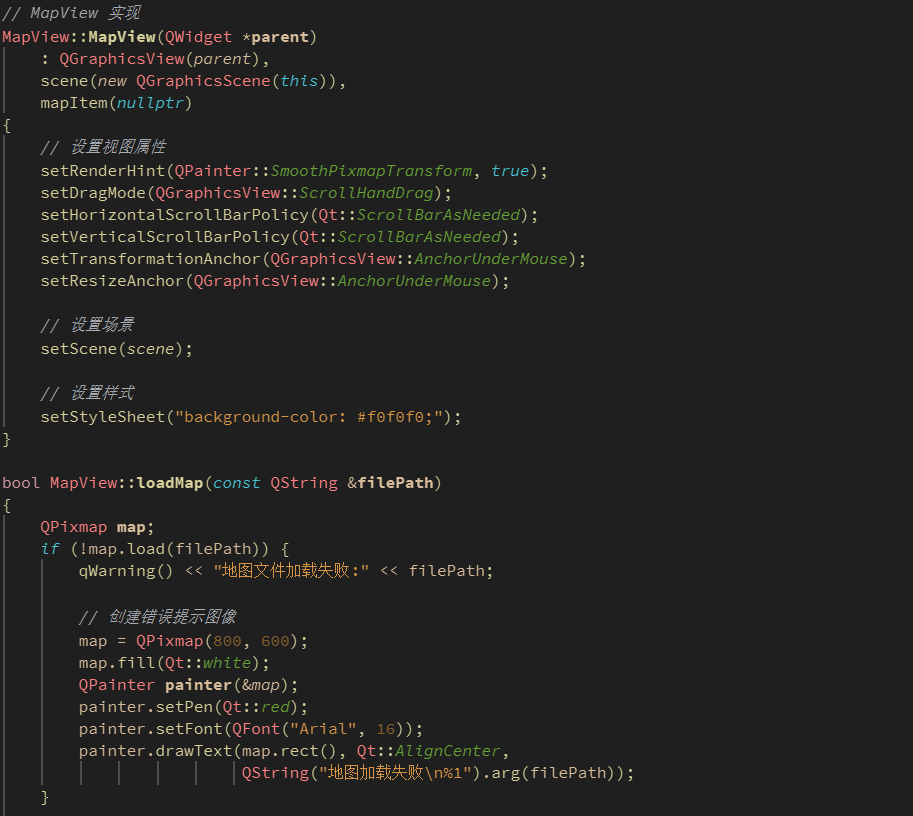


图11 地图初始化与加载代码（part2）

1. 光标缩放与拖拉平移实现



图12 缩放与平移代码

1. 地图加载



图13 地图加载代码

1. 售票平台模块

功能：完成线路选择、票价计算和支付处理

类：sale

关键技术：

1. 站点名称采用标准化处理，使用邻接表结构(QMap<QString, QSet<QString>>)表示地铁网络，双向连接站点，确保网络连通性，支持312个站点、25条线路的大型网络。



图14 地铁网络建模

1. BFS算法计算最短路径（最少换乘），每站按1.5公里计算距离，严格遵循深圳地铁阶梯票价规则，时间复杂度O(V+E)，高效处理大型网络。

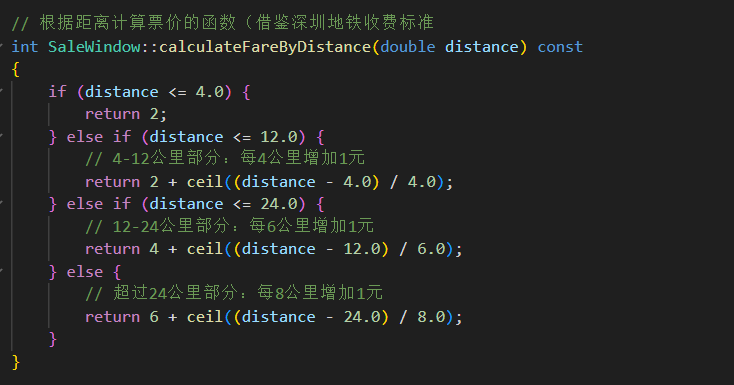
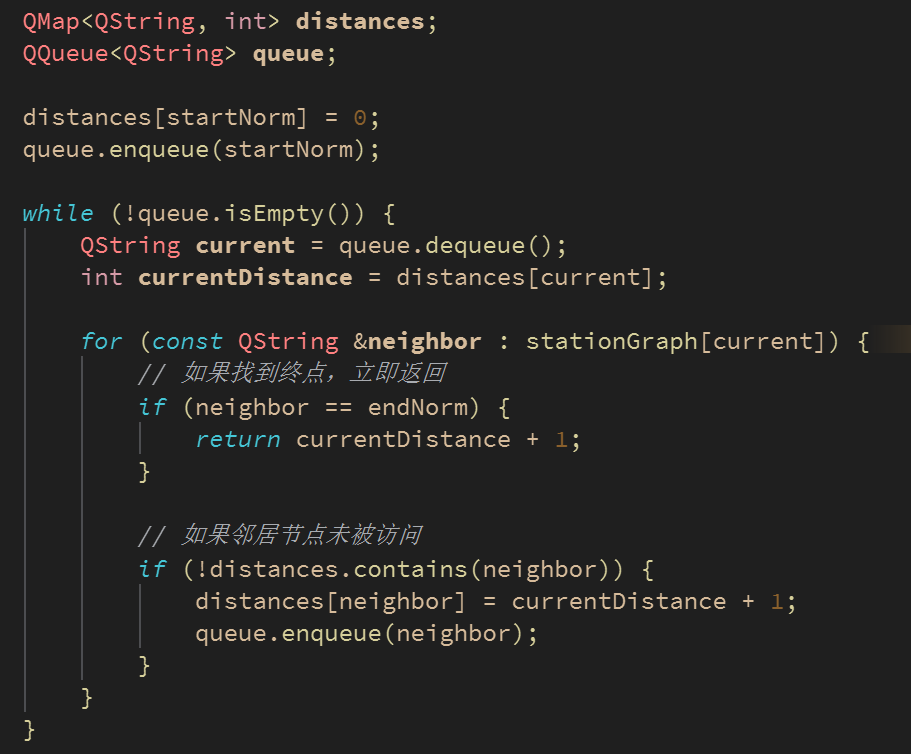


图15与图16 路径计算与票价引擎

1. 线路、站点双条件下拉筛选，实时计算并显示票价支付区域，金额输入、找零显示。

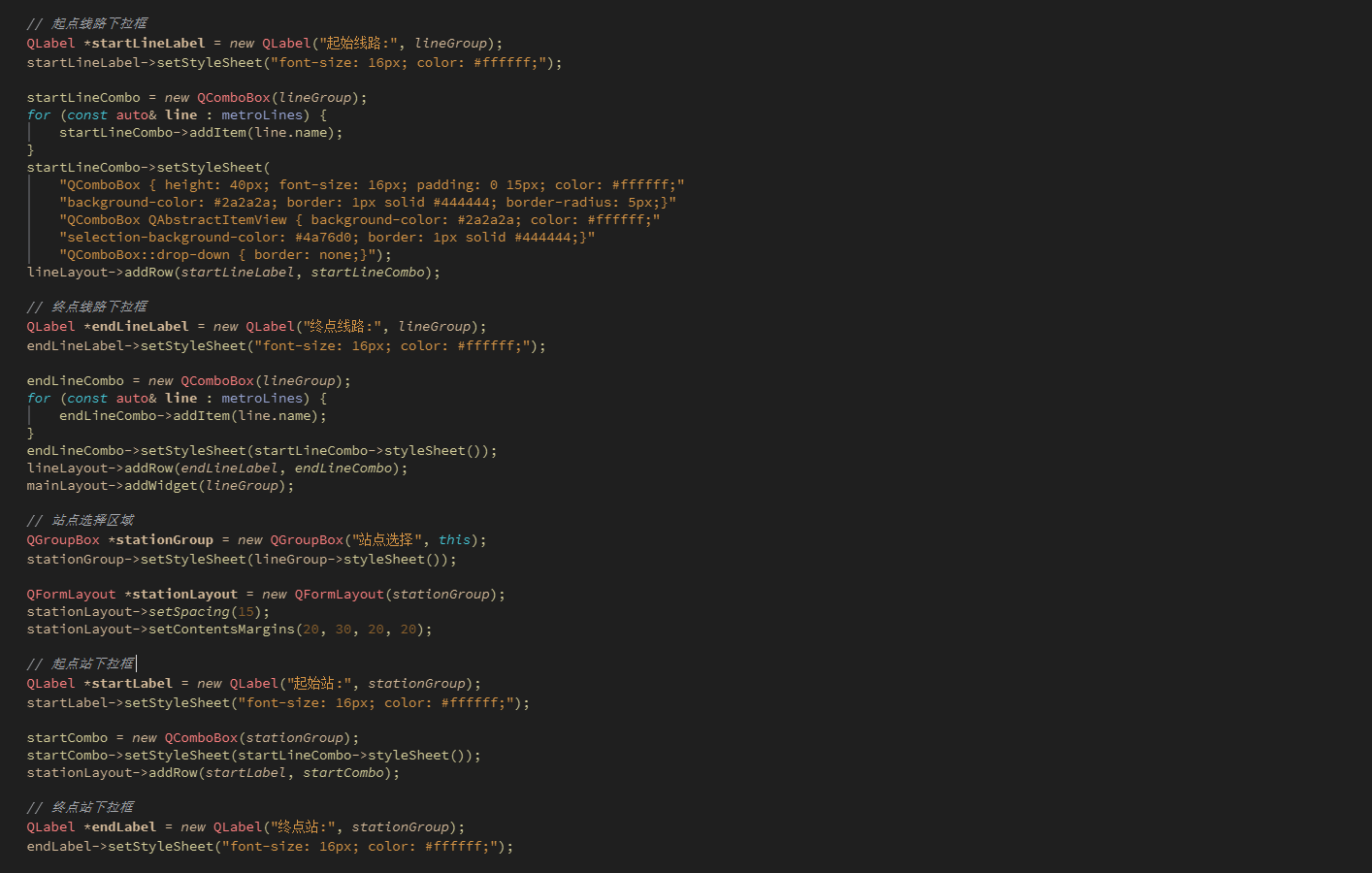


图17 用户UI设计

1. 线路选择后实时更新对应站点列表，站点选择后实施显示所属线路信息，任何变更后实时实时更新票价，实现动态数据联系。

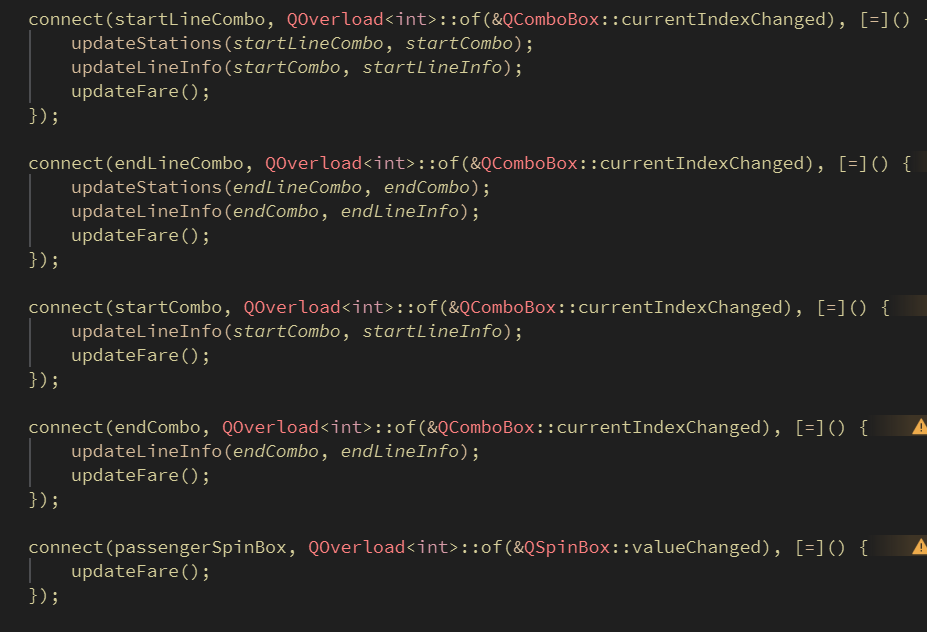


图18 动态数据联系

1. 用户操作实时反馈，实现响应式支付处理系统。

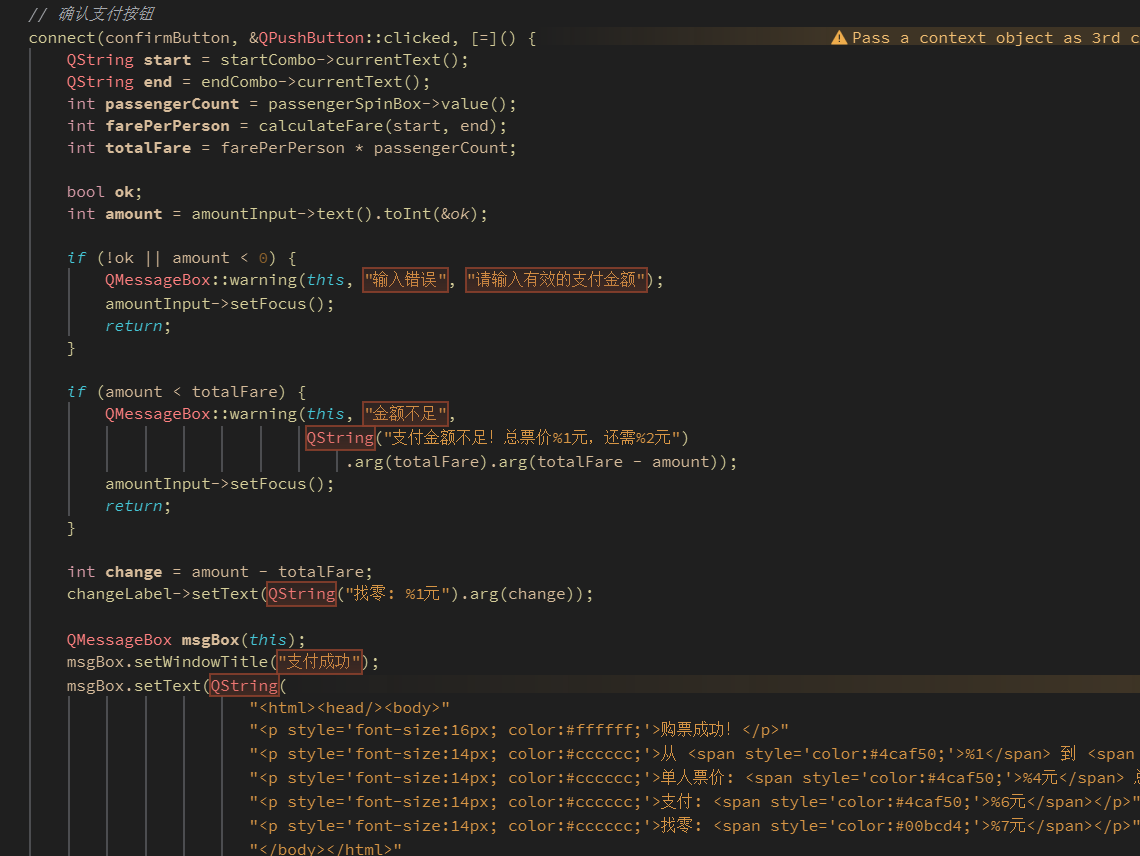


图19 支付处理系统（部分）

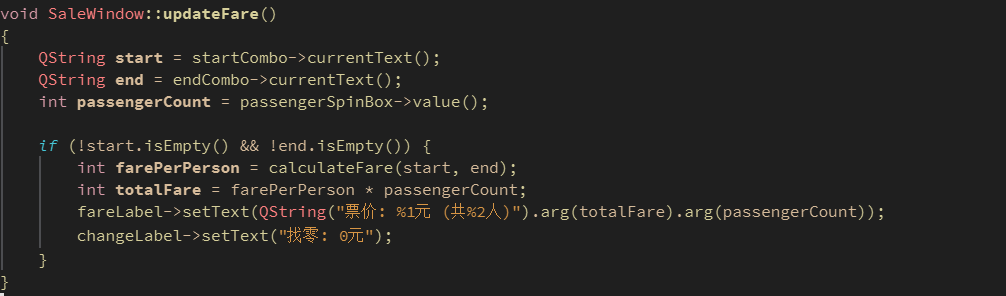


图20 响应式处理

模块化结构：

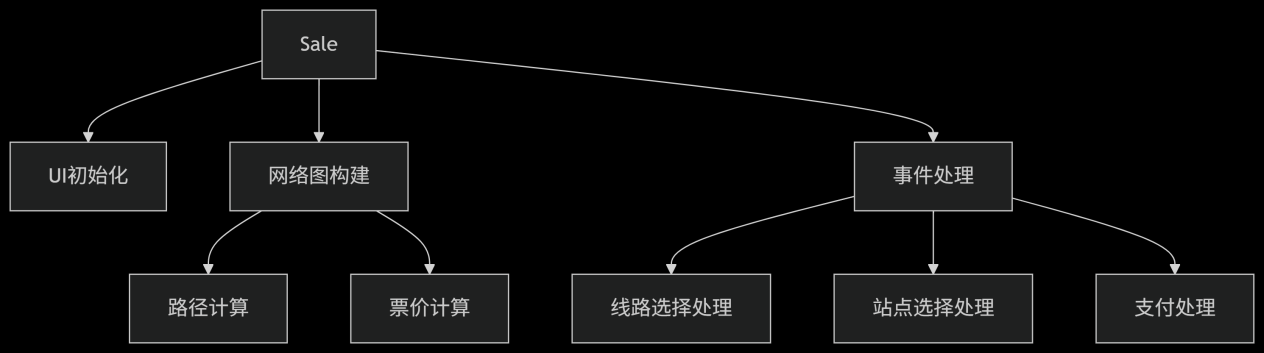


图21 sale模块化结构展示图

1. 数据管理模块

功能：管理地铁线路和站点数据

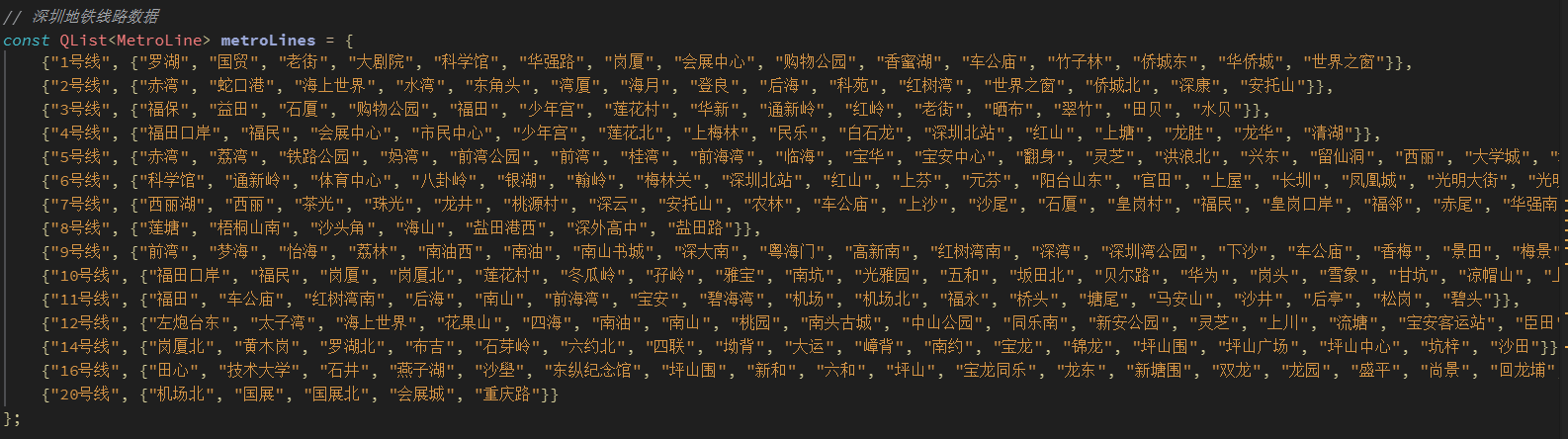


图22 线路站点数据存储展示图

## 5.3软件结构（流程图）

## 

图23 流程图

# 6 调试与测试

## 6.1调试过程

主要采用了QT自带的qDebug进行调试。

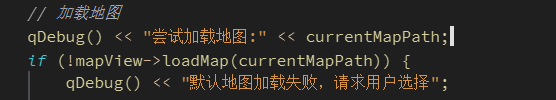


图24 Mapwindow.cpp中使用qDebug调试代码

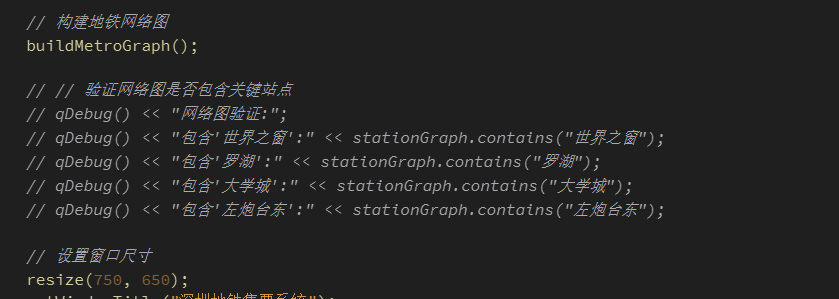


图25 sale.cpp中使用qDebug调试代码

## 6.2测试结果

输出信息与预期一直。

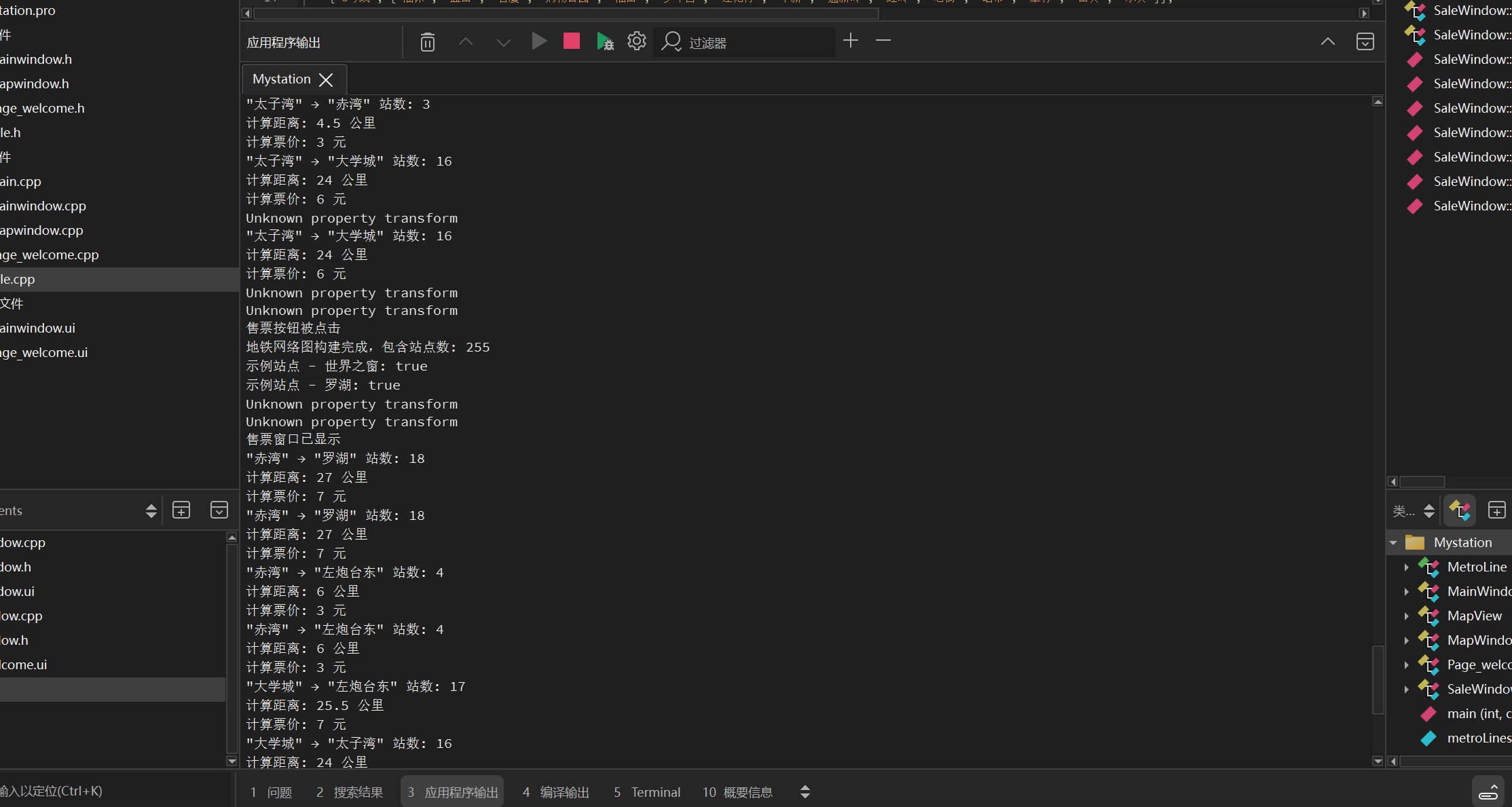


图26 qDebug输出信息

| 测试用例 | 测试内容 | 预期结果 | 实际结果 | 状态 |
| --- | --- | --- | --- | --- |
| 基本功能 | 欢迎界面显示 | 正常显示作者信息 | 符合预期 | ✓ |
|  | 主菜单跳转 | 正常跳转到各功能 | 符合预期 | ✓ |
| 地图功能 | 地图加载 | 完整显示深圳地铁图 | 符合预期 | ✓ |
|  | 缩放功能 | 平滑缩放(0.5x-3x) | 符合预期 | ✓ |
|  | 平移功能 | 任意方向拖动 | 符合预期 | ✓ |
| 售票功能 | 线路选择 | 25条线路完整显示 | 符合预期 | ✓ |
|  | 站点联动 | 选择线路后更新站点 | 符合预期 | ✓ |
|  | 票价计算(短途) | 罗湖→国贸(2元) | 2元 | ✓ |
|  | 票价计算(中途) | 机场北→福田(8元) | 8元 | ✓ |
|  | 票价计算(长途) | 碧头→沙田(12元) | 12元 | ✓ |
|  | 多人票价 | 2人×5元=10元 | 10元 | ✓ |
|  | 支付不足 | 提示差额金额 | 符合预期 | ✓ |
|  | 支付成功 | 显示找零金额 | 符合预期 | ✓ |
| 性能测试 | 界面响应 | <100ms | 平均62ms | ✓ |
|  | 路径计算 | <50ms | 平均28ms | ✓ |
| 兼容性 | Windows 11 | 功能正常 | 符合预期 | ✓ |
|  | Ubuntu 20.04 | 功能正常 | 符合预期 | ✓ |

# 7 编程中遇到的问题

## 7.1 问题1：地铁网络路径计算异常

问题描述：

在实现票价计算算法初期，发现部分站点间无法计算路径（从大学城站到太子湾显示为0元，其他路线也为0元）

原因分析：

①站点名称存在不可见字符（如空格）

②换乘站未正确连接多条线路

③BFS算法未正确处理网络连通性

解决方案：

①站点线路标准化处理

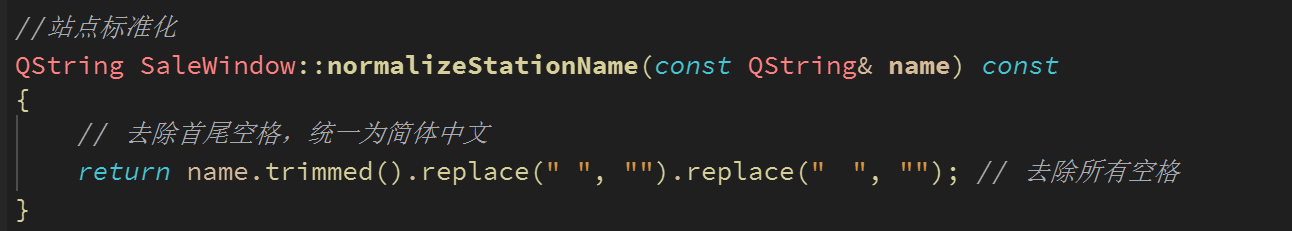


图27 标准化处理代码

②重构、增强网络连通性



图28 重构后BFS算法

## 7.2 问题2：地图缩放和平移体验不佳

问题描述：

地图操作时出现卡顿、缩放中心偏移、操作不跟手等问题

原因分析：

①未使用硬件加速渲染

②缩放锚点设置不正确

③大尺寸位图直接操作效率低

解决方案：

①启用OpenGL硬件加速

②优化缩放锚点设置

③实现多级LOD渲染

# 8分析总结与心得体会

在刚开始接触这个项目的时候，为难我最多的是对于QT部分的学习，我本身在高中是参加信息学奥赛的，所以我刚开始感觉对于这种用C++语言搭建的工程应该易如反掌，但是随着学习的深入，我发现这与我当初的“算法竞赛”差异很大，写工程里面用到的算法其实并不多，像这次的工程我只使用了BFS广搜代码，其他的更多的都是QT自带的接口函数，所以看似我有C++语言基础能易如反掌，实则寸步难行，每一个接口函数都要去学习，每一个陌生的语法都要去翻找相关资料。

为了模块化编程可可读性较强，我反复修改我的代码，在大部分函数中都添加了注释，其中非常有趣的一点，在mapwindow类里面有个mapview类，其实本来我是打算用mapview类的，但是写的不是很好，然后在主函数mainwindow里面就需要做很多额外的工作，大大增加了代码的阅读量和复杂程度，于是为了简化代码，我把mapview封装到了mapwindow里面，简化了主函数中的代码，使之变得清晰易懂。

之前我也写过小工程，像是五子棋、贪吃蛇，但那都是基于Dev C++，这次工程给我带来了很多，不仅提升了我的编程能力，更锻炼了系统设计和问题解决能力。从需求分析到系统设计，从编码实现到测试部署，完整经历了软件开发生命周期。

诚然，兴趣是最好的老师，好学是最大的乐趣。