# 长沙理工大学

## 《专业方向综合实训》报告

# 基于 QT 的车载系统的设计与实现


| | | | |
|---|---|---|---|
| 学　　院 | 计算机与通信工程 | 专　　业 | 计算机科学与技术 |
| 班　　级 | 计科 18 卓越 | 学　　号 | 201850080221 |
| 学生姓名 | 张仁华 | 指导教师 | 汤强 |
| 课程成绩 | | 完成日期 | 2021.7.9 |

# 计算机工程实训成绩评定

学　　院 计算机与通信工程　专　　业 计算机科学与技术

班　　级 计科 18 卓越　学　　号 201850080221

学生姓名 张仁华　指导教师 汤强

课程成绩 _____　完成日期 2021.7.9

## 指导教师对学生在实训的评价

| 评分项目 | 优 | 良 | 中 | 及格 | 不及格 |
|---|---|---|---|---|---|
| 实训中的创造性成果 | | | | | |
| 学生掌握课程内容的程度 | | | | | |
| 实训完成情况 | | | | | |
| 实训动手能力 | | | | | |
| 文字表达 | | | | | |
| 学习态度 | | | | | |
| 规范要求 | | | | | |
| 实训论文质量 | | | | | |

## 指导教师对实训的评定意见

综合成绩 _____指导教师签字_____　　年　月　日

# 基于 QT 的车载系统的设计与实现

## 摘要

车载系统（AutomoTIve OperaTIng System，简称 AOS）是管理和控制车载硬件与车载软件资源的程序系统，是直接运行在 AB 上的最基本的系统软件，任何上层软件，HMI，数据连接都必须在操作系统的支持下才能运行。

车载操作系统是用户和车载硬件的接口，同时也是车载硬件和上层软件的接口。车载系统的功能包括管理车载系统的硬件、软件及数据资源，控制程序运行，改善人机界面，为上层软件提供支持，让车载系统的资源，以及接收到数据、信号、音频、视频最大限度地发挥作用，提供各种形式的用户界面（UI），使驾驶员有一个好的驾驶体验，并有效的提供辅助驾驶、半自动驾驶、甚至自动驾驶。

作为一个基本车载系统的设计，该车载系统实现了四项基本功能，包括音乐播放，视频播放，天气查询，地图查询。　。

**关键词**：车载系统；QT；嵌入式

# DESIGN AND IMPLEMENTATION OF QT-BASED SYSTEM

## ABSTRACT

**As a basic vehicle-mounted system design, my vehicle-mounted system implements four basic functions, including music playback, video playback, weather query, and map query. As an in-vehicle system, its design is convenient and the starting point for everything is designed for the convenience of users.AutomoTIve OperaTIng System (AOS) is a program system that manages and controls vehicle hardware and vehicle software resources.**

**The vehicle operating system is the interface between the user and the vehicle hardware, controlling program operation, improving the man-machine interface, providing support for the upper software, allowing the vehicle system resources, and receiving data, signals, audio, and video Give full play to its effect, provide various forms of user interface (UI), so that the driver has a good driving environment, and effectively provide assisted driving, semi-automatic driving, and even automatic driving.**

**Keywords: vehicle-mounted system; QT; embedded**

# 目　录

# 1 绪论

对软件需求的透彻理解，对于软件开发工作能否成功是起着至关重要的作用，需求说明的任务是发现、规范的过程，这有益于提高软件开发过程中的能见度，便于对软件开发过程中的控制与管理，有助于采用工程方法开发软件，提高软件的质量，还有利于开发人员、维护人员、管理人员之间的交流以及协作，并且可以作为工作成果的原始依据：再者，亦可在向潜在用户传递软件功能、性能需求，使其能够判断该软件是否与自己的需求相符合。

## 1.1 项目的研究背景

与如今竞争激烈的快节奏时代中，随着人口基数的上涨，人们的生活水平越来越繁忙。因为此，愈来愈多的人感到内心的空虚与压力带来的烦躁，这些也带给人们许许多多心理和身体上的困惑。所以如何缓解当代人们的生活压力，已成为人们的共同夙愿。汽车的使用可谓是人手一台，各行各业都在为汽车的便利性作优化作更好的服务，以便在市场上占据有利的主导地位。

## 1.2 项目的目的和意义

在此车载系统中搭配了四种功能，足够满足人们平时出行所需要的基本保障所需。通过点击车载按钮即可达到播放音乐，观看视频的娱乐满足，同时也可通过天气模块获得及时的天气状况以便作下一步的安排，还可以通过地图模块查询实时地图动态，规划出行路线。

本设计意义在于帮助开车的人在驾驶过程中不那么枯燥无趣，利用在车上开车的时间打开音频或者视频播放器获得身心上的愉悦体验，达到放松的状态，从而使开车过程更加流畅舒适，减少驾驶员的身心疲惫，有利于降低车祸事件的发生。同时在驾驶员规划出行方案和出行路线时，可以利用天气预报模块查看最近一周内所有的天气情况，同时通过百度地图模块规划出行路线防止驾驶员驶离错误方式。本系统既放松了驾驶员的身心，又方便了驾驶员的出行。

## 1.3 项目的主要任务

采用结构化的设计方法，开发出一个操作简单，UI 界面直观明了，方便驾驶员操作的车载系统，并逐个实现各个模块的功能。

车载系统功能需求即主要任务如下图 1.1 所示：

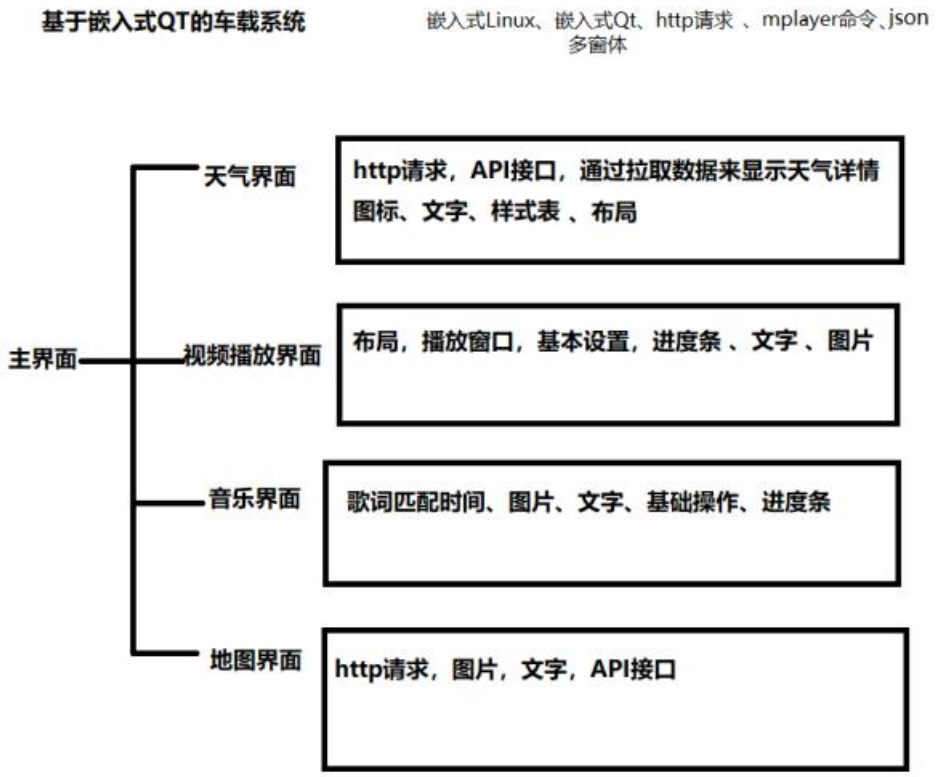



图 1.1 车载系统工能需求

# 2 相关技术介绍

## 2.1 Qt

QT 是诺基亚开发的跨平台 C++图形用户界面应用程序开发框架。它既可以开发 GUI 程序，也可用于开发非 GUI 程序，比如控制台工具和服务器。Qt 是面向对象的框架，使用特殊的代码生成扩展以及一些宏，易于扩展，允许组件编程。它提供给应用程序开发者建立艺术级的图形用户界面所需的所用功能。1996 年后，QT 进人商业领域，它已经成为全世界范围内数千种成功的应用程序的基础。QT 也是流行的 Linux 桌面环境 KDE 的基础。基本上，QT 同 X Window 上的 Motif, Openwin, GTK 等图形界面库和 Windows 平台上的 MFC, OWL, VCL, ATL 是同类型的东西，但 QT 具有优良的跨平台特性、面向对象、丰富的 API、大量的开发文档等优点。

信号和槽机制是 QT 的核心机制，信号和槽是一种高级接口，应用于对象之间的通信，它是 QT 的核心特性，也是 QT 区别于其它工具包的重要地方。Qt 利用信号与槽机制取代传统的 callback 来进行对象之间的沟通[1]。当操作事件发生的时候，对象会发提交一个信号；而槽则是一个函数接受特定信号并且运行槽本身设置的动作。信号与槽之间，则通过 QObject 的静态方法 connect 来链接。

同时，Qt 也是一个跨平台的 C++图形用户界面库，由挪威 TrollTech 公司出品，目前包括 Qt/X11， 基于 Framebuffer 的 Qt Embedded，快速开发工具 Qt Designer，国际化工具 Qt Linguist 等，Qt 支持 Unix 系统及 Linux，还支持 WinNT/Win2k，Win95/98 平台。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 API 和开发工具对所有支持平台都是一致的，从而可以进行独立于平台的程序开发和配置。它使得跨平台软件编程直观、简易和方便。Qt 提供了一种称为 signals/slots 的安全类型来替代 callback 回调函数，这使得各个控件之间的协同工作变得十分简单。

## 2.2 C++

C++是 C 语言的继承，它既可以进行 C 语言的过程化程序设计，又可以进行以抽象数据类型为特点的基于对象的程序设计，还可以进行以继承和多态为特点的面向对象的程序设计。C++擅长面向对象程序设计的同时，还可以进行基于过程的程序设计，因而

C++就适应的问题规模而论，大小由之[2]。在完成 C++标准化的第一个草案后不久，发生了一件事情使得 C++标准被极大地扩展了：Alexander stepanov 创建了标准模板库（Standard Template Library，STL）。STL 不仅功能强大，同时非常优雅，然而，它也是非常庞大的。在通过了第一个草案之后，委员会投票并通过了将 STL 包含到 C++标准中的提议。STL 对 C++的扩展超出了 C++的最初定义范围。虽然在标准中增加 STL 是个很重要的决定，但也因此延缓了 C++标准化的进程。委员会于 1997 年 11 月 14 日通过了该标准的最终草案，1998 年，C++的 ANSI/IS0 标准被投入使用[3]。通常，这个版本的 C++被认为是标准 C++。所有的主流 C++编译器都支持这个版本的 C++，包括微软的 Visual C++和 Borland 公司的 C++Builder。

# 3 系统总体设计与环境配置

## 3.1 车载系统主界面模块

本车载系统的主要功能模块主要分为五个模块，分别是音乐播放模块，视频播放模块，天气查询模块，地图查询模块，以及主界面。主要实现了各个模块内的功能需求以及主界面的自由切换。

本项目由我自己一人独立完成。

如图 3.1 所示，车载系统主界面有四个按钮可以进入到不同的模块中去，点击模块即可进入。

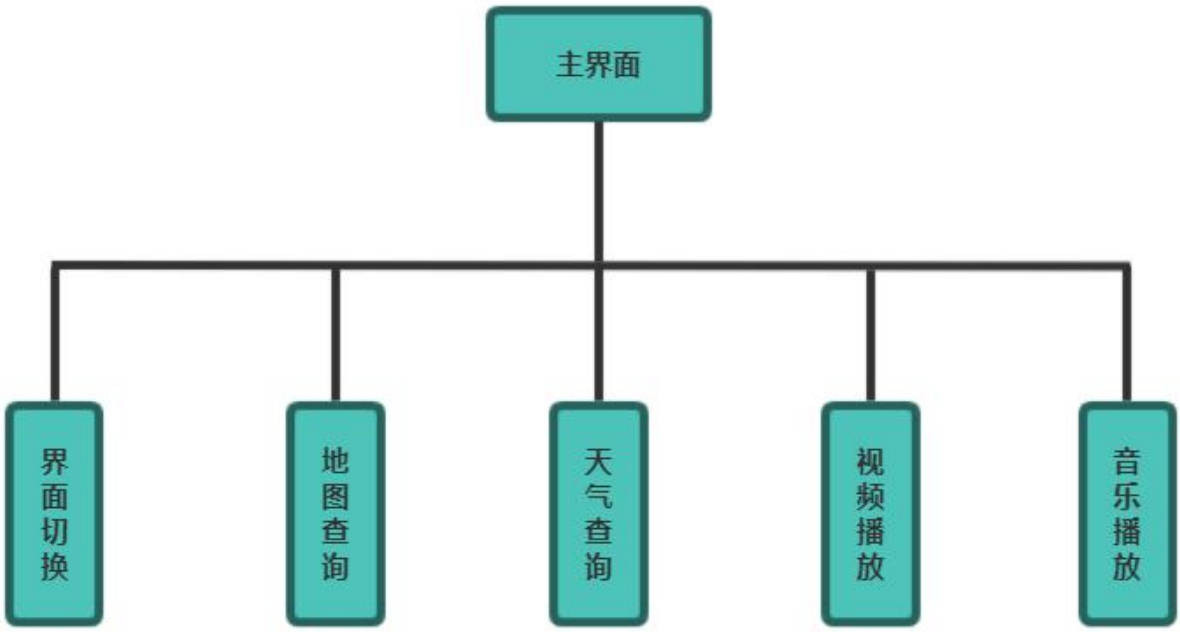



图 3.1 主界面总体设计

## 3.2 音乐播放模块

如图 3.2 所示进入音乐模块对音乐的播放进行选择，如果按下播放按钮则播放音乐，再次按下播放按钮暂停音乐。停止播放时可按下上/下按钮进行音乐的选择。当需要返回主界面按下返回按钮即可。

音乐播放器最基础的功能，播放与调节，音乐选择，音量调节，返回按钮等等是我要实现的。歌词不作设计，如果显示歌词，会分散驾驶员的注意力，不利于健康安全的车辆行驶。

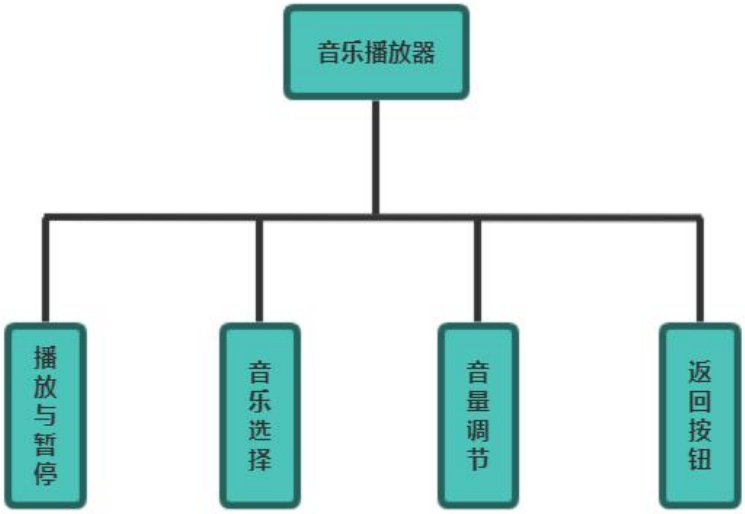



图 3.2 音乐模块设计

## 3.3 视频播放模块

如图 3.3 所示进入视频模块列表中是没有文件的，需要打开本地文件夹导入本地的视频文件到播放列表中去。通过上/下选择想要观看的视频，同时可以通过点击快进与快退调节视频的进度。通过点击静音按钮实现音频的关闭，打开声音调节滑动状态条可调节音量的大小。

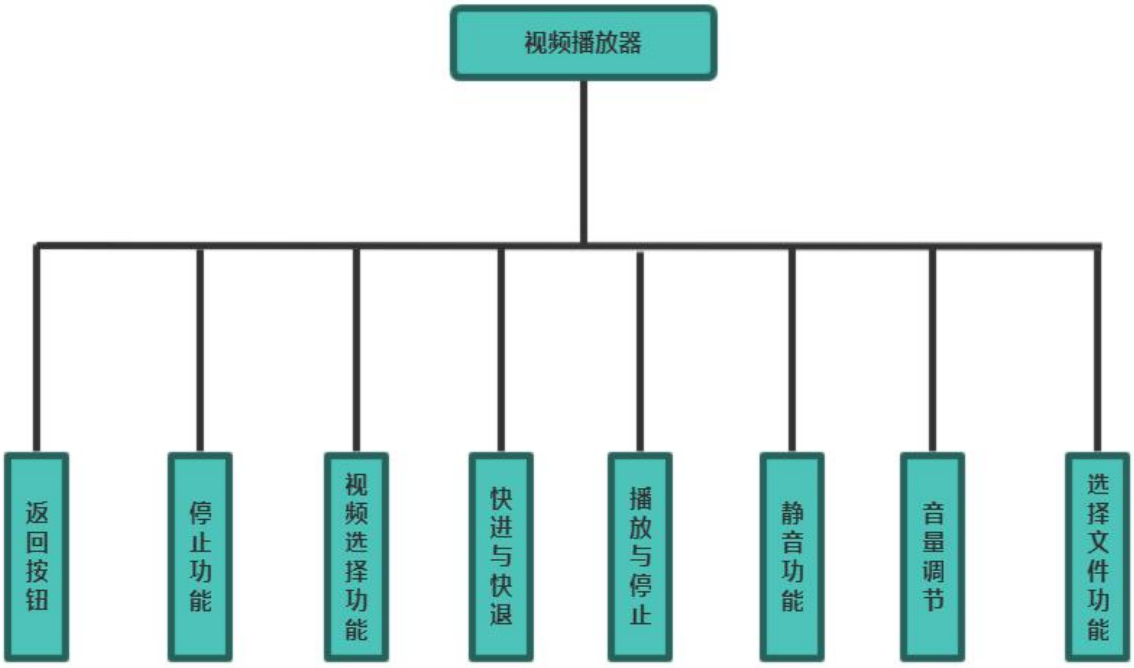


图 3.3 视频模块设计

## 3.4 天气查询模块

如图 3.4 进入天气模块后需要对城市名，天气情况，天气图片，更新时间，空气质量指数，级别，风速，湿度以及未来一周的天气预测进行显示。利用 API 接口得到自己需要的天气数据，进行一定的处理过后，按照既定格式进行排版显示。同时找好每个天气需对应的图片，显示不同天气时 UI 界面也需要进行相应的变换处理。

需按照图 3.4 相应的功能进行严格的流程设计，每一个功能都需逐步实现，显示完整的天气预报。

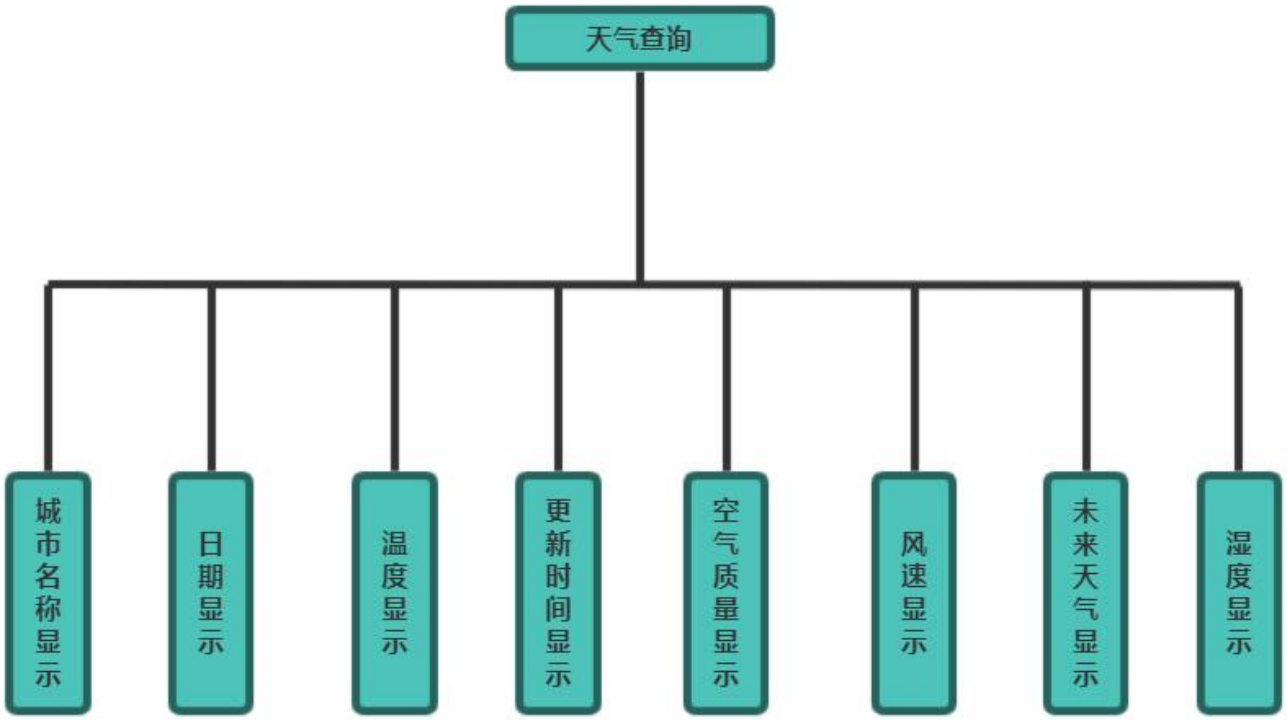


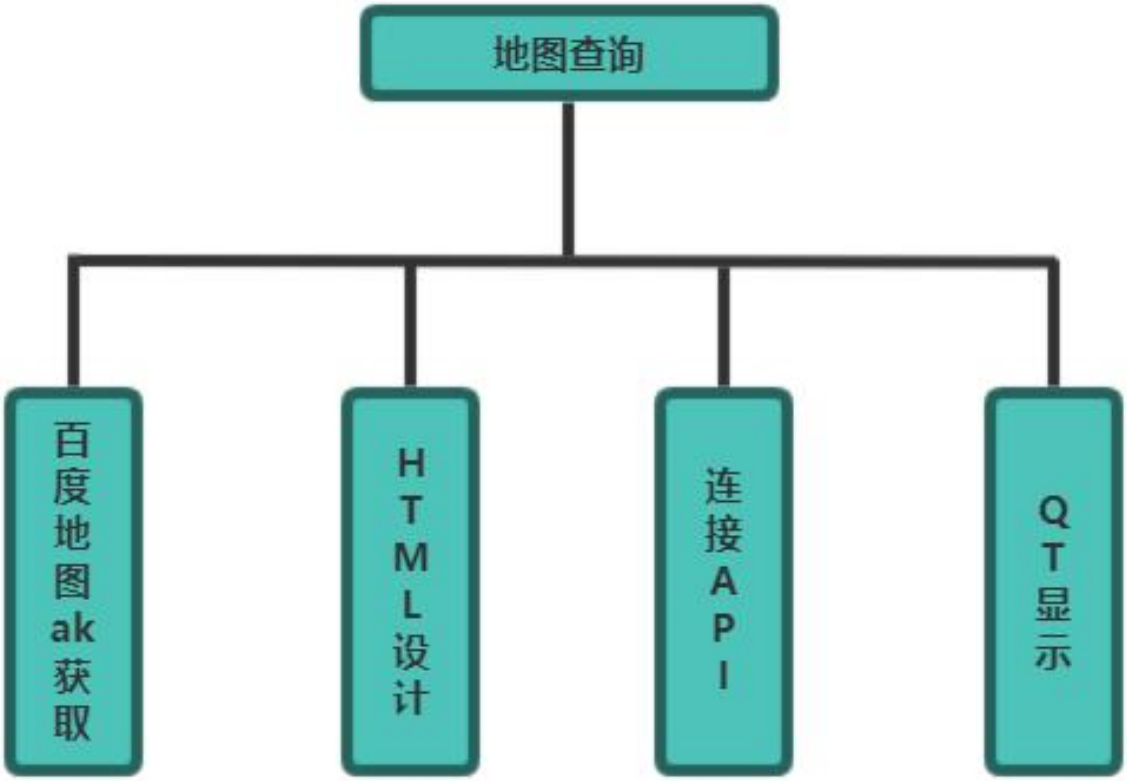

图 3.4 天气查询模块设计

## 3.5 地图查询模块

如图 3.5 地图查询模块的主要设计从获取百度地图的 ak 密匙，从而通过连接百度地图的 api 获取实时的地图数据，其中最重要的一环就是 html 的设计，在 html 设计中加入鼠标滚轮的缩放，还有可视化缩放控件，移动控件，比列尺显示，三种不同地图的显示与切换。

百度地图 JavaScript API 嵌入到 QT 中，需提前做好 HTML 地图页面，同时也需考虑 HTML 和 QT 控件兼容性的问题。

地图显示需要 QT 与 JS 交互，这里需要注意 QT5.6 以后的版本交互可以使用 runJavaScript 函数来进行，这里安装的 QT 版本是 5.12。




3.5 地图查询模块

# 4 车载系统详细设计

## 4.1 主界面实现

### 4.1.1 主界面设计思路

UI 设计中分别构建音乐，视频，天气，地图四个 PushBotton 按键，通过 SLOT 点击事件分别进入到不同的模块中。同时在各个模块中添加返回按钮 back_btn，通过在各子界面点击 back_btn 返回到主界面中

### 4.1.2 UI 设计实现

定义四个 PushButton 按钮，分别命名为 map_bt,music_bt,video_bt,weather_bt,同时添加时间 Lable 命名为 time 显示实时时间。通过改变样式表对各个控件进行美化和优化，其中时间 Lable 样式 background-color: transparent 设置，使它处于透明状态，无法看见。如下图 4.1 所示。

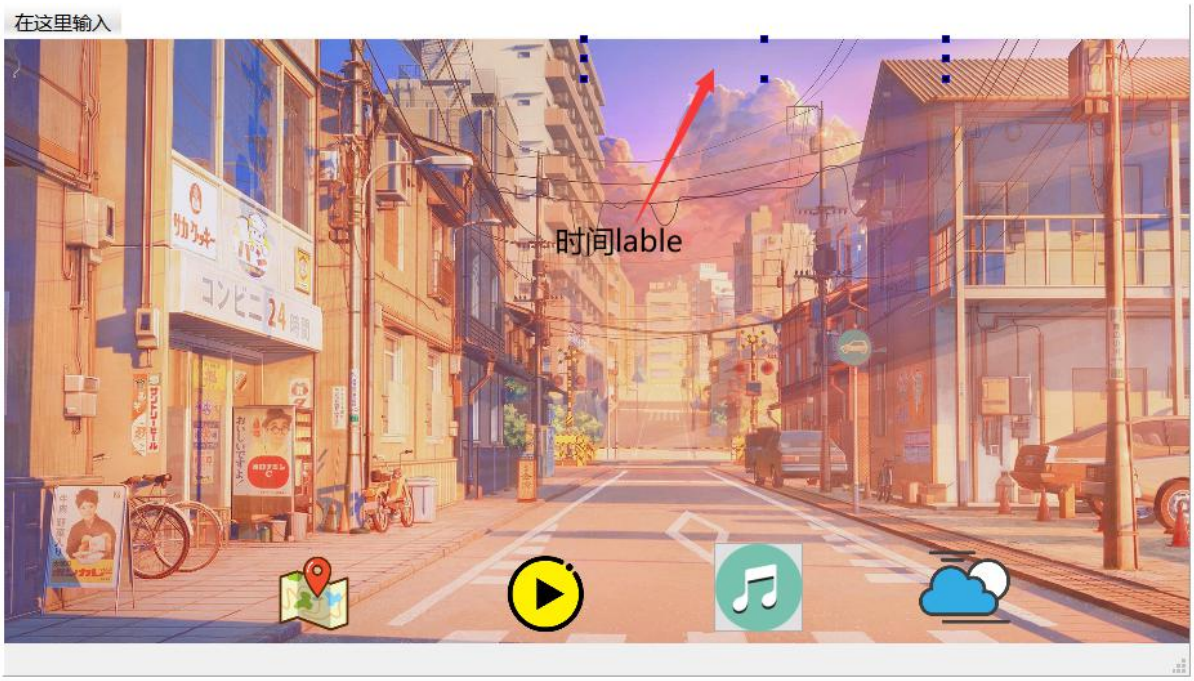



图 4.1.主界面 UI 设计

### 4.1.3 主界面功能实现

1.定义四个子界面窗口

```
music = new musicwindow(this);
video = new videowindow(this);
map = new mapwindow(this);
weather = new weatherwindow(this);
```

2.点击 clicked( )传递信号调用 show( )函数显示子窗口，同时调用 hide( )函数隐藏主界面。

```
music->show();
this->hide();
```

3. 自定义 getTime( )函数获取本地时间，通过 setText( )设置到 Lable 中去。

```
QDateTime time =QDateTime::currentDateTime();
QString str= time.toString("yyyy-MM-dd hh:mm:ss dddd");
ui->time->setText(str);
```

## 4.2 音乐播放实现

### 4.2.1 音乐播放设计思路

QT 播放音乐需要调用 QMediaPlayer 这个类，通过这个类的 paly( ),pause( )函数实现播放与暂停，同时这个类还包装了很多有关的方法供开发者使用。

音乐播放模块 UI 设计使用了 ListWidget 来显示音乐列表，用四 PushBotton 实现各个功能。

通过转到槽点击事件控制音乐播放器。

### 4.2.2 UI 设计实现

相定义四个 PushButton 按钮，分别实现播放，暂停，上一曲，下一曲，声音音量显示，定义 Slider 控件滑动控制音量的大小。对界面样式进行优化过后得到图 4.2 。

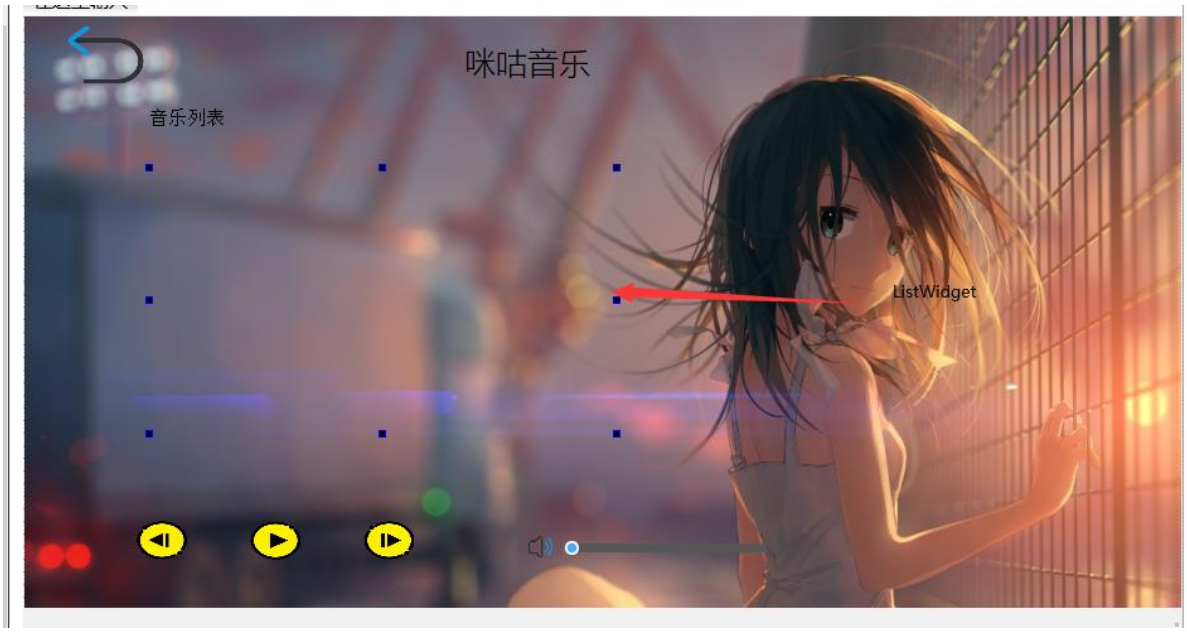



图 4.2 音乐播放界面 UI 设计

### 4.2.3 音乐播放界面功能实现

1. 读取文件显示到 ListWidget 中，通过自定义函数 readFile( )实现，通过设定指定音乐文件夹路径，把后缀为 .mp3 的文件储存到 List 中去，再通过 ListWidget 控件 addItems 方法把 List 中的文件加入到 ListWidget 中，最后通过调 sortItems( ) 方法显示音乐列表

```
QStringListfiles=dir.entryList(nameFileters,QDir::Files|QDir::Readable,
        QDir::Name);
ui->list_music_2->addItems(files);
ui->list_music_2->sortItems();
```

2.实例化对象 QMediaPlayer，通过 setMedia( )函数，使指针指向音乐列表，实现上一曲，下一曲的功能，同时使用 play( ),pause( )函数实现对音乐的播放与暂停。

```
p_player = new QMediaPlayer(this);
p_player->setMedia(QUrl::fromLocalFile(str_pathmusic +'/'+str_music));
```

3.函数 setVolume( )可调节音量。

## 4.3 视频播放实现

### 4.3.1 视频播放设计思路

QT 播放视频同样可以使用 QMediaPlayer 这个类，实现对视频的调用，播放和各种各样的操作。

UI 界面使用 ListWidget 显示视频播放列表，VideoWidget 作为视频窗口进行播放，通过 PushBotton 控件转到槽 SLOT 事件对视频进行控制。

### 4.3.2 UI 设计与实现

界定义九个 PushBotton 按钮分别实现返回主界面，快进与快退，切换上一个与切换下一个，播放与暂定功能，静音功能，音量控件 Slider 的显示。一 VIdeoWidget 和一个 ListWidget 分别显示视频和视频播放列表，整个布局如图 4.3 所示。

图 4.3 视频播放 UI 界面

### 4.3.3 视频播放界面功能实现

1. 返回主界面，调用父类 show 函数显示 mainwindow，同时使用 hide 函数隐藏本界面。

```
this->parentWidget()->show();
this->hide();
```

2. 通过使用 setCurrentRow 函数获得视频进度条的位置以便实现鼠标点击进度条位置对视频进行拖动。改变播放位置使用 setRange 函数和 positionchanged 函数使视频进度条和视频播放同步。

```
ui->horizontalSlider->setRange(0,duration);
ui->horizontalSlider->setValue(position);
```

3. 自定义播放函数 play，当点击播放按钮时 play 函数自动获取当前状态值，如果处于播放状态，则返回 1，调用 pause 函数使视频暂停，反之调用 MidiaPlayer 类的 play 函数使视频进行播放。

```
switch (mediaplayer->state()) {
case QMediaPlayer::PlayingState:
    mediaplayer->pause();
    break;
default:
     mediaplayer->play();
break;
```

4. 点击打开文件按钮，显示添加文件，通过设置文件过滤器只读 wmv，mp3，avi 为后缀的视频文件。如果选中一个或多个文件，则返回的 count>=1，进入到 for 循环中逐步添加到容器中，最后通过调用 addItem 函数显示到界面中去。

```
if (fileList.count()<1)
     return;

for (int i = 0; i<fileList.count();i++)
 {
    QString aFile=fileList.at(i);
    playlist->addMedia(QUrl::fromLocalFile(aFile));
   QFileInfo fileInfo(aFile);
    ui->listWidget->addItem(fileInfo.fileName());
 }
```

5. 调用 previous 函数指向上一个视频。

```
    playlist->previous();
```

6. 调用 next 函数指向下一个视频。

```
    playlist->next();
```

7. 获取进度条现在的位置，定义 t 为当前位置，自定义点击快进时往后退两秒，则 t+=2000，使得进度条的位置往后挪两秒，反正 t-=2000，使进度条的位置往前挪两秒，实现快进与快退。

```
qint64 t;
t = mediaplayer->position();
t += 2000;
mediaplayer->setPosition(t);    //快进
```

```
qint64 t;
t = mediaplayer->position();
t -= 2000;
mediaplayer->setPosition(t);  //快退
```

8. 调用 setMuted 函数实现静音功能

```
mediaplayer->setMuted(!mute);
```

9. 音量点击显示按钮和音量调节拖动与音乐播放界面一模一样。

## 4.4 天气播放实现

### 4.4.1 天气查询设计思路

QT 获取天气信息一般都是通过调用天气服务器的接口来获取的，网上的 API 分为两种，一种是 XML 编码格式的天气信息，一种是 Json 编码格式的天气信息，这里我选用解析 Json 编码格式的天气信息 API。

通过 QT 的 HTTP 通信，获取天气 API 的内容，再进行解析，实现连续一周的天气显示；实现对长沙的天气信息的查询（长沙城市 ID：101250101）

### 4.4.2 UI 设计

Q 页面分为上下两个板块，整个页面的控件均由 Lable 组成，上面部分界面显示的是长沙市今天最近一个时间点更新的详细天气状况，包括日期，时间，温度，天气状况，空气指数及其级别，风速，湿度；下面的界面显示的未来六天内长沙市的天气情况，最高温度还有最低温度。如图 4.4 所示。

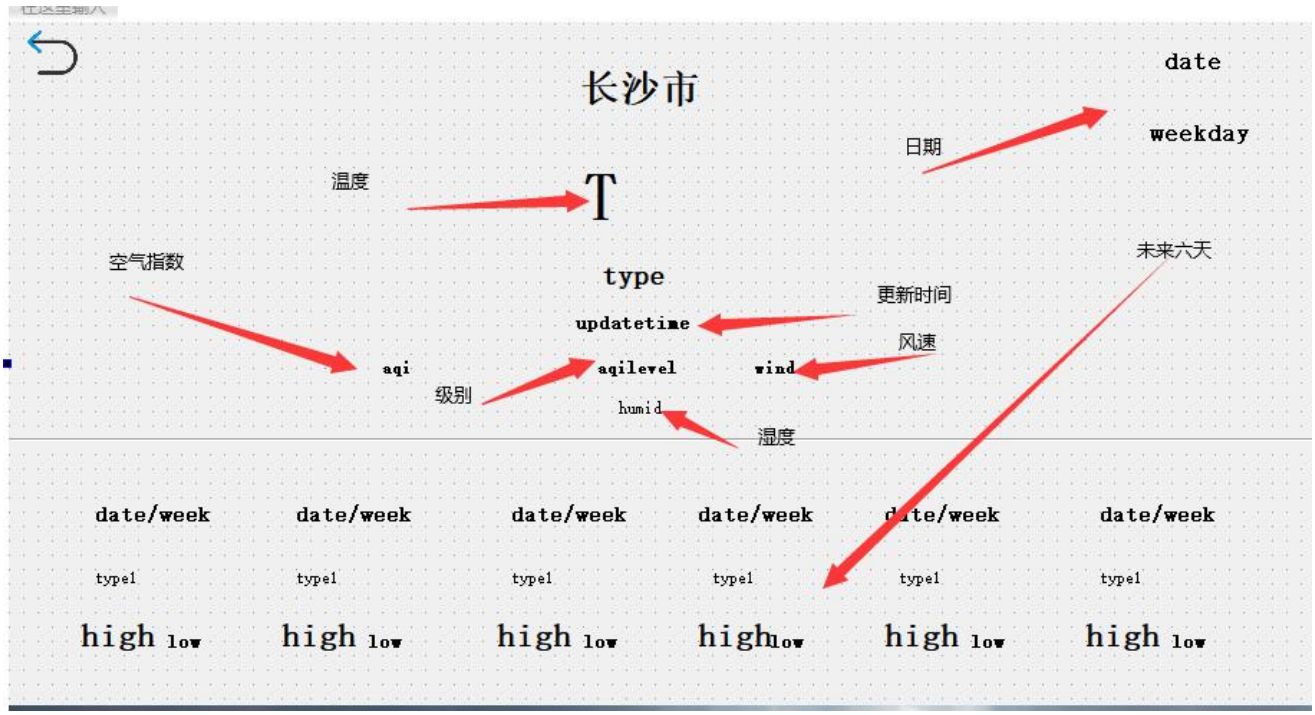



图 4.4 天气 UI 界面设计

### 4.4.3 天气查询功能实现

1.返回主界面，和前面的方法一样

2.调用对象方法 get/post 发送请求（附带一个请求对象），选择相应的 URL，这里我只选用了长沙市。

```
connect(manager, SIGNAL(finished(QNetworkReply *)),this,
    SLOT(replyFinished(QNetworkReply *)));
manager->get(QNetworkRequest(QUrl("http://t.weather.itboy.net/api/weather/city/101250101")));
```

3.把得到的 Json 进行解读，Json 格式的数据是以“{}”包含，Qt 把这整体数据叫做“JsonDocument”，相当于目录，并且提供了 QJsonDocument 类来将信息转换为 Qt 能够解析的 Json 数据格式,和 QJsonParseError 类来检测数据是不是 Json 格式。“{}”里面存放的是键值对，键值对格式为 key:value  key 为键，value 为值，value 可以是对象，数组，数，字符串。

（1）判断是否为 json 格式的数据，如果不是 json 格式，则程序自动退出

```
QJsonDocument json = QJsonDocument::fromJson(js.toUtf8(),&error);
if(error.error != QJsonParseError::NoError)
  {
      qDebug()<<error.errorString();
   }
else
      qDebug()<<"init json succeed!";
```

（2）Json 格式一层一层解析

① 如果键对应的值为数或字符串时，里面的内容叫做“QJsonObject”，可以由 QJsonDocument 的“object()”方法来获取

```
//对象里的 data
QJsonValue value1 = obj.take("data");
QJsonObject obj1=value1.toObject();

QJsonValue value2 = obj.take("cityInfo");
QJsonObject obj2=value2.toObject();

//cityInfo 里的 city
QJsonValue value_city = obj2.take("city");
ui->city->setText(value_city.toString());
```

② 如果键对应的值为数组时，里面的内容叫做“QJsonArray”，再通过遍历将数组将数据取出来(我这里的 json 数据里面的数组都是存放的键值对，且键值对存放的内容为字符串)

```
for(int i=0;i<6;i++)
 {
   QVariantMap map = list[i].toMap();
   dw[i]->setText(map["date"].toString()+"("+map["week"].toString()+")");
   hg[i]->setText(map["high"].toString().mid(3).remove("℃")+"° ");
   lw[i]->setText(map["low"].toString().mid(3).remove("℃")+"° ");
```

4. 根据得到天气设置相应的图片到 UI 上，如果是晴天，则设置 sunny.png，如果是多云则设置 cloundy.png 等等

```
if(map["type"].toString()=="晴")
ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/sunny.png")));

if(map["type"].toString()=="多云")
ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/cloudy.png")));

if(map["type"].toString()=="阴")
ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/yintian.png")));

if(map["type"].toString()=="小雨")
ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/xiaoyu.png")));

if(map["type"].toString()=="霾")
ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/mai.png")));
```

## 4.5 地图播放实现

### 4.5.1 地图查询设计思路

利用 QT 的 QAxWidget 控件实现程序内嵌百度地图，涉及到 Qt 后台与百度地图前端的数据交互以及函数调用。利用百度地图给出的开发手册和免费的 API 写出百度地图的 JavaScrip，然后将编写好的 HTML 文件嵌入到 QT 文件中去。

### 4.5.2 UI 设计

一共三个控件，一个返回按钮，一个显示 HTML 路径，一个显示内嵌的地图。如图 4.5 所示。

返回按钮转到槽，返回上一个界面。

要使用 QAxWidget 控件，需要在 car.pro 中加入 QT+= axcontainer，重新构建即可。QT 直接支持 ActiveX 对象，所以在 QT 中可以直接使用 QAxWidget，也就是窗口控件对象。例如 word, excel,powerpoint,pdf,flash,html 等。

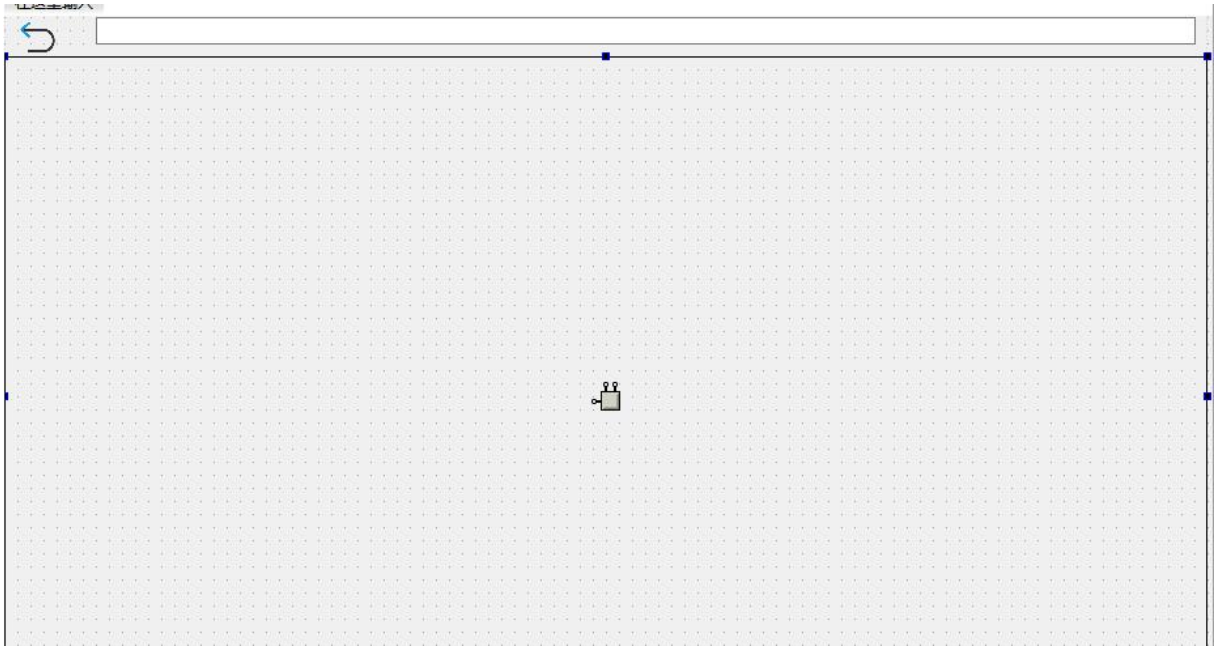


图 4.5 地图 QT 界面

### 4.5.3 百度地图 HTML 设计

1. 使用百度地图 JavaScript API，获取 ak 码。如图 4.6 所示。


图 4.6

2.HTML 设计，设置容器样式，创建地图元素。

```
<style type="text/css">
        body, html{width: 100%;height: 100%;margin:0;font-family:"微软雅黑";}
        #allmap{height:500px;width:100%;}
        #r-result{width:100%; font-size:14px;}
</style>
```

3. 初始化地图并显示。

```
var map = new BMap.Map("allmap",{
        minZoom : 1,
        maxZoom : 20
```

```
});
//初始化地图，设置坐标点在哪
map.centerAndZoom(new BMap.Point(116.404323,39.905201),18);
```

4.设置自定义控件，这里我添加了比列尺控件，移动控件，缩放控件，鼠标滚动控件。

```
map.centerAndZoom(new BMap.Point(116.404323,39.905201),18);
map.enableScrollWheelZoom(true);
map.addControl(new BMap.NavigationControl());
map.addControl(new BMap.ScaleControl());
map.addControl(new BMap.OverviewMapControl());
map.addControl(new BMap.MapTypeControl());
```

### 4.5.2 地图显示功能实现

1.获取地图 HTML 所在位置路径，写入是 sFilePath 中

```
ui->lineEdit->setText(sFilePath);
ui->axWidget->setControl(QString::fromUtf8("{8856F961-340A-11D0-A96B-00C04FD705A2}"));
```

2. 自定义 loadNavigate 函数在 QAxWidget 实现 QT 与百度地图 JavaScript API 的交互。

```
QString Utr = ui->lineEdit->text().trimmed();
ui->axWidget->dynamicCall("Navigate(const QString&)",Utr);
```

# 5 测试

## 5.1 主界面测试

首先运行 QT，点击主界面图 5.1 所示，分别进入到四个不同的界面如图 5.2, 5.3, 5.4, 5.5 所示。点击各个页面的返回按钮能返回到主界面。

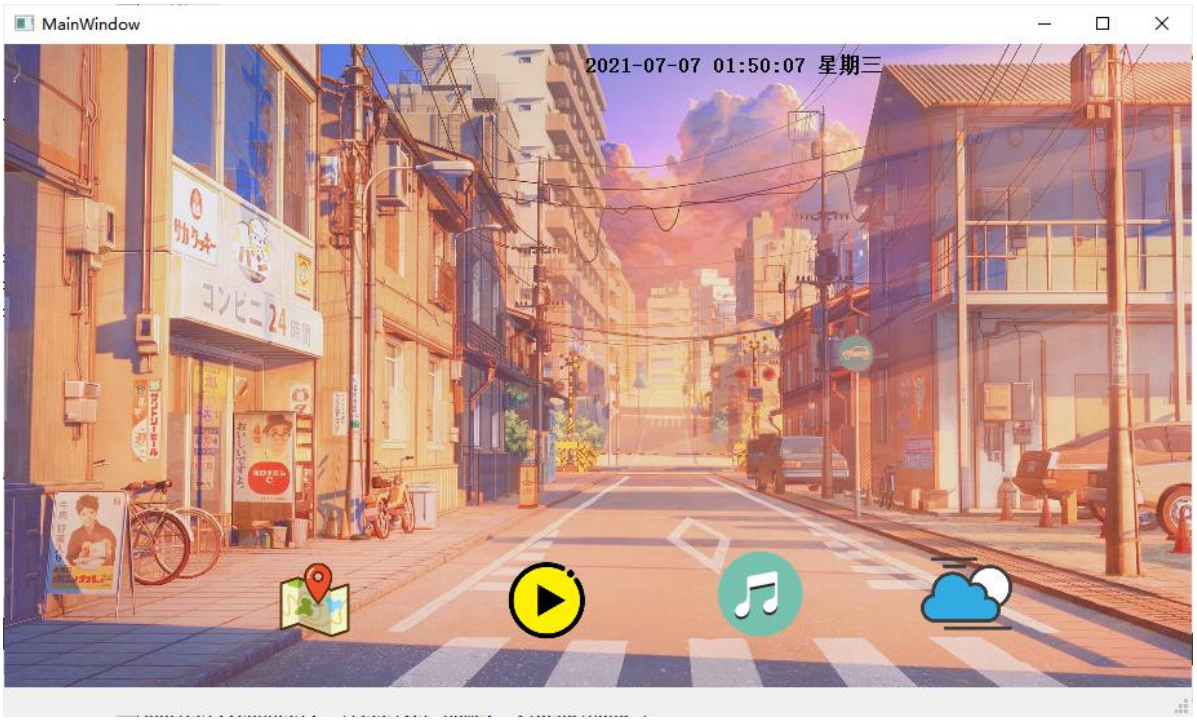



图 5.1

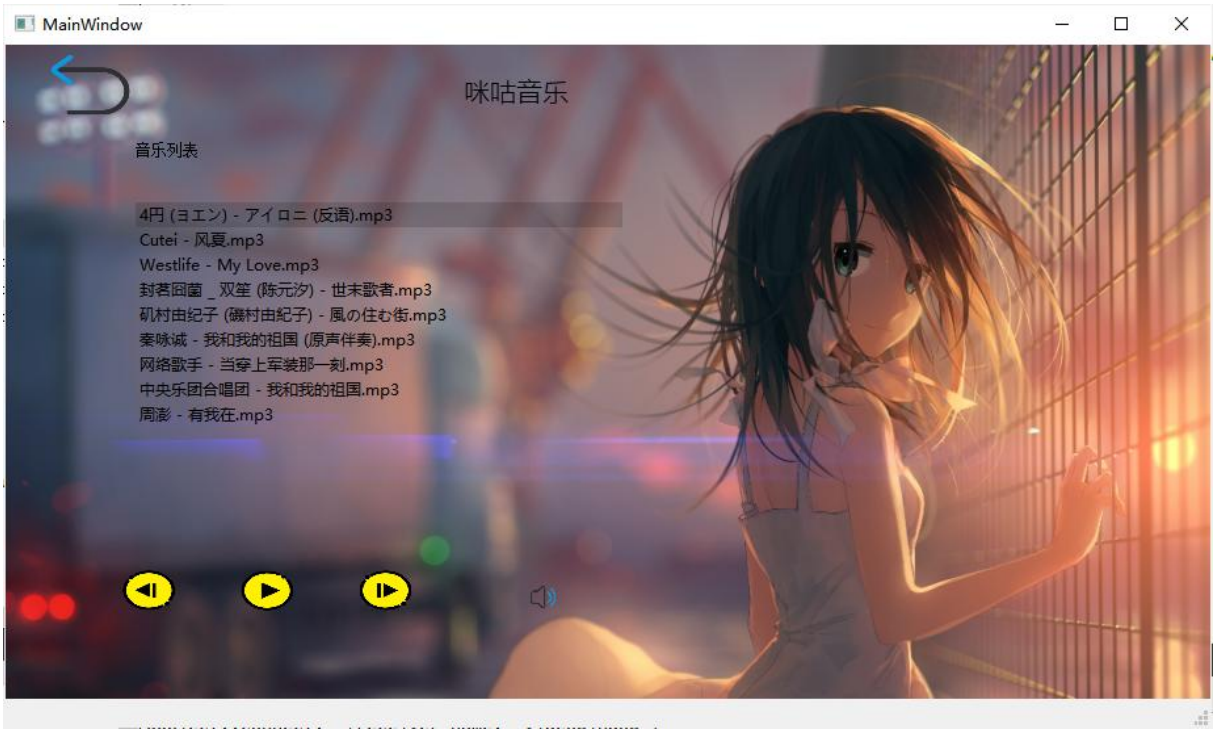



图 5.2

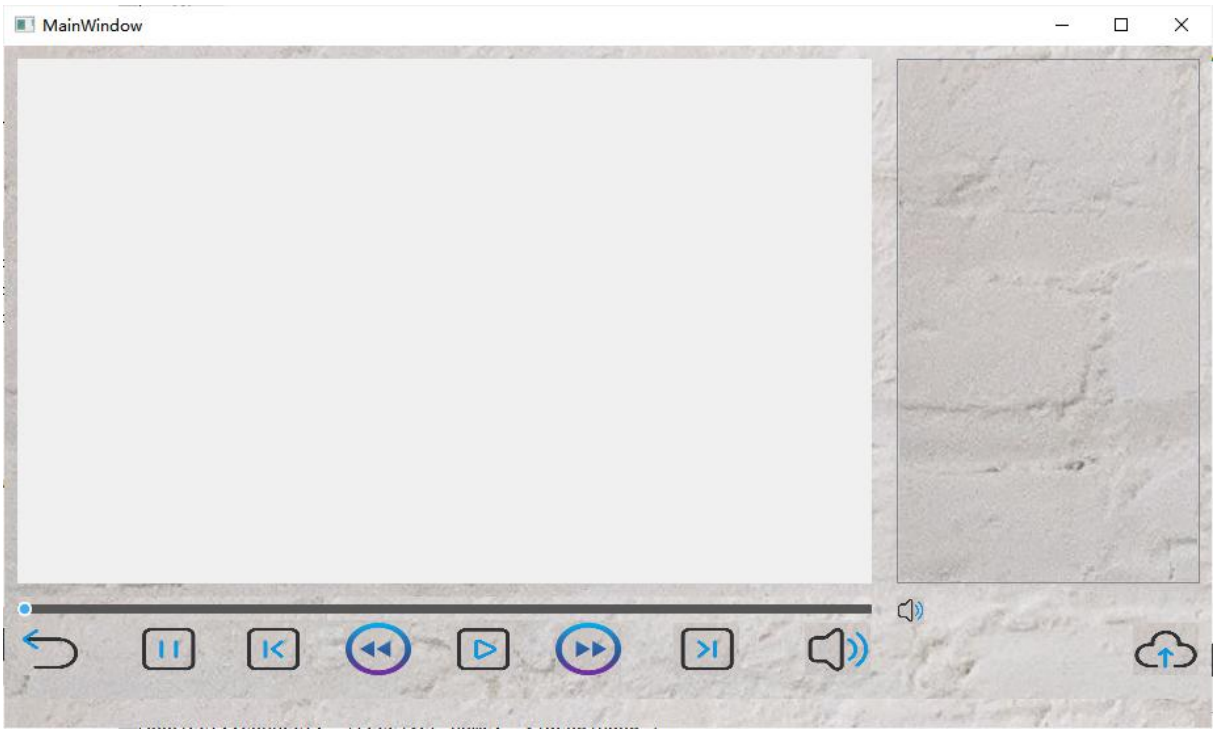

MainWindow

图 5.3


MainWindow
长沙市
2021-07-07
星期三
33℃
晴
更新时间 00:16
空气质量指数 30
级别 优
风速 南风 2级
湿度 84%
07(星期三)
08(星期四)
09(星期五)
10(星期六)
11(星期日)
12(星期一)
37° 28°
37° 28°
37° 28°
37° 28°
37° 28°
36° 28°

图 5.4

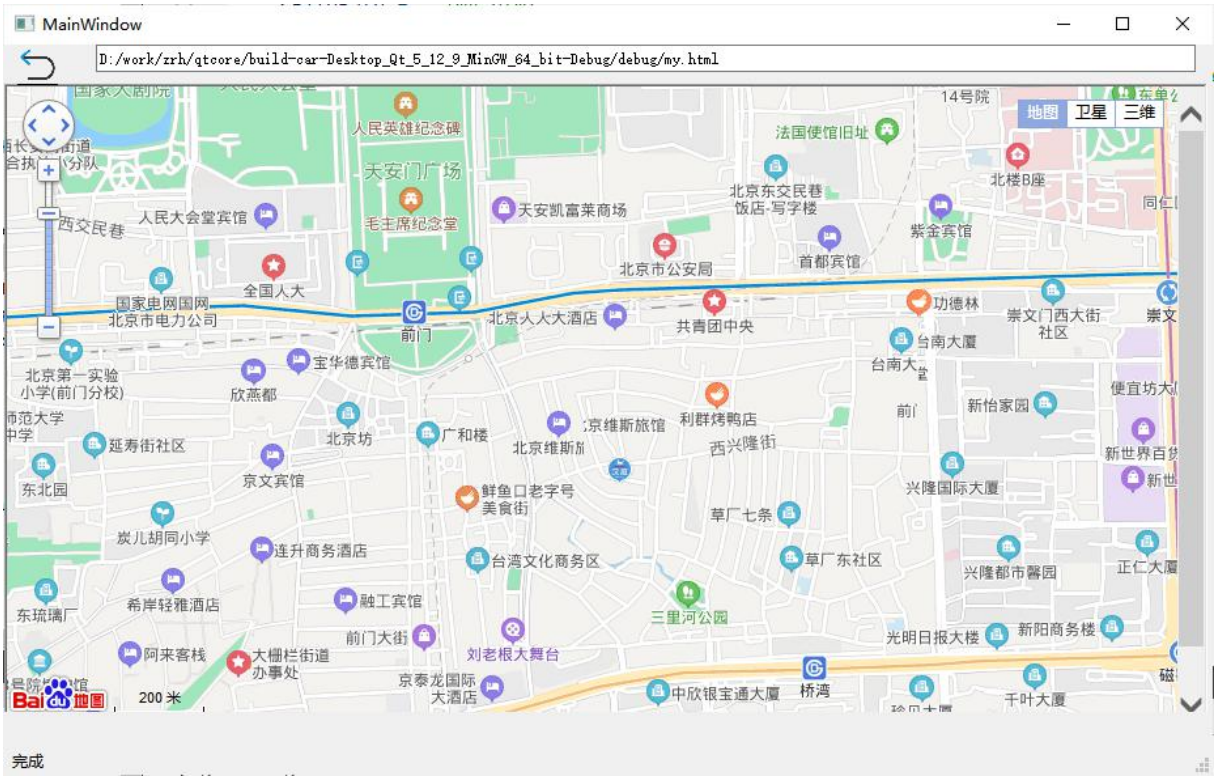



图 5.5

## 5.2 音乐播放测试

（1）点击播放，按钮会变成 pause 样式。如图 5.6。


（2）点击上一曲/下一曲 会进行歌曲的选择。如图 5.7。

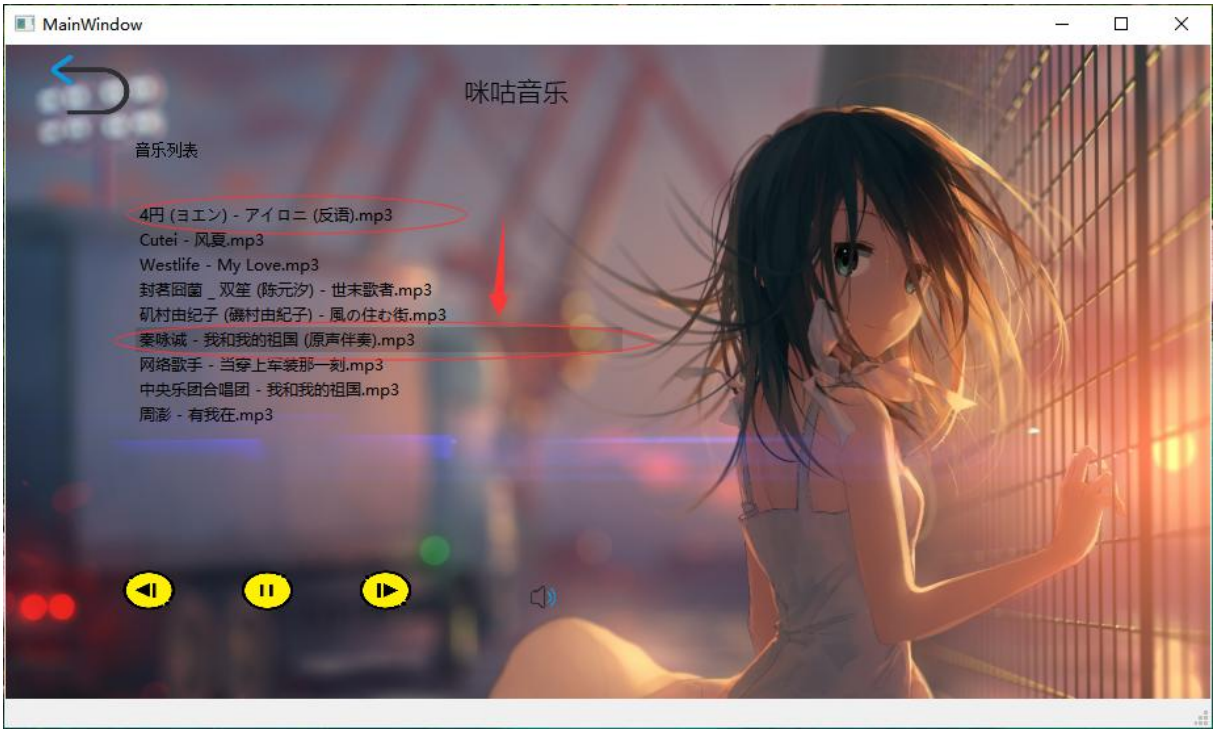



图 5.7

（3）按下音量键会显示音量滑动条，通过滑动可调节音量的大小。 如图 5.8

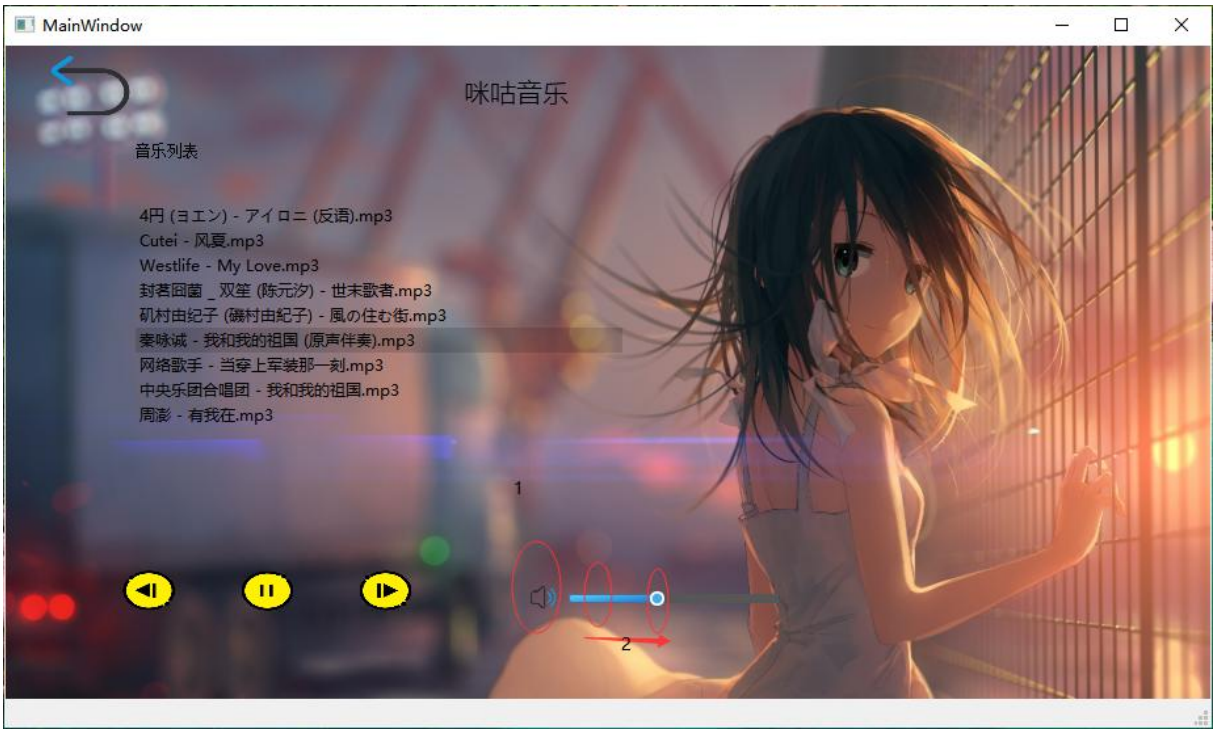



图 5.8

## 5.3 视频播放测试

（1）打开文件会弹出添加文件信息，同时进行筛选，添加好的文件会显示到列表中。如图 5.9,5.10

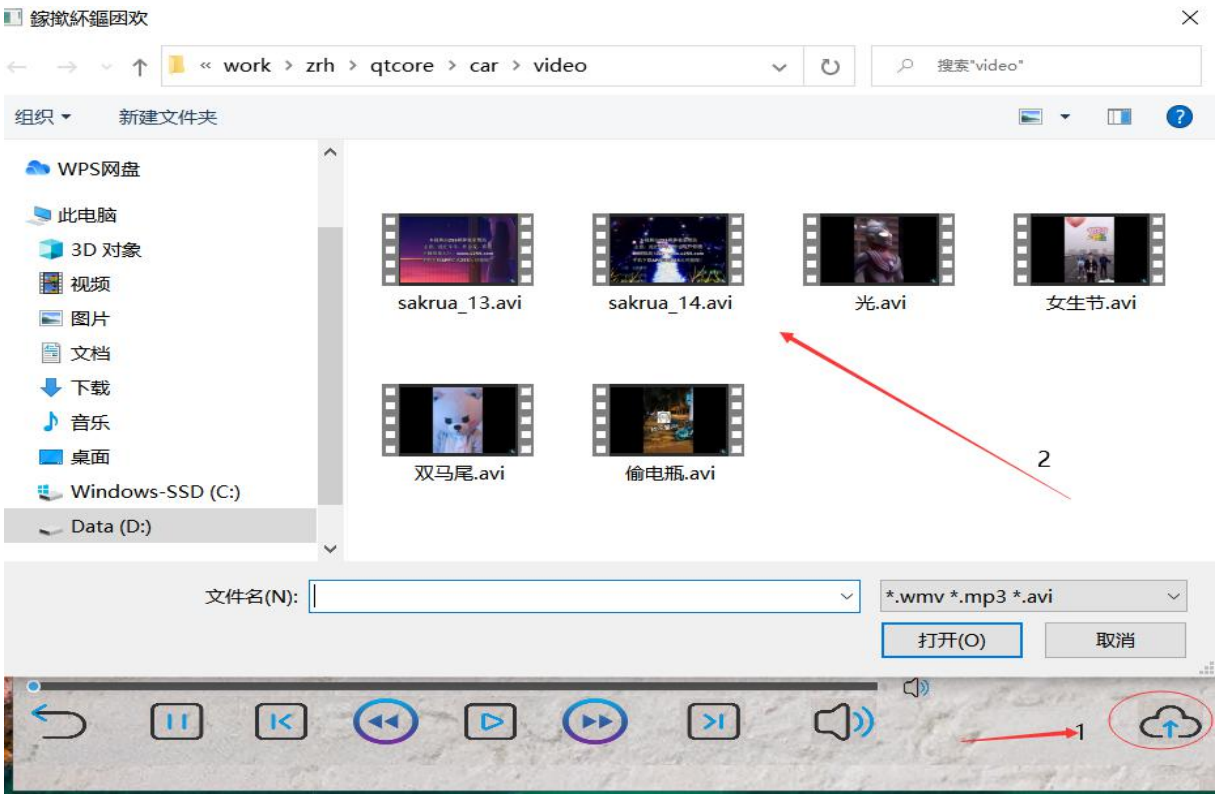



图 5.9


图 5.10

（2）按下播放按钮视频会停止播放。如图 5.11


图 5.11

（3）按下停止按钮则会使视频回到最初的位置，这和暂停不一样。如图 5.12

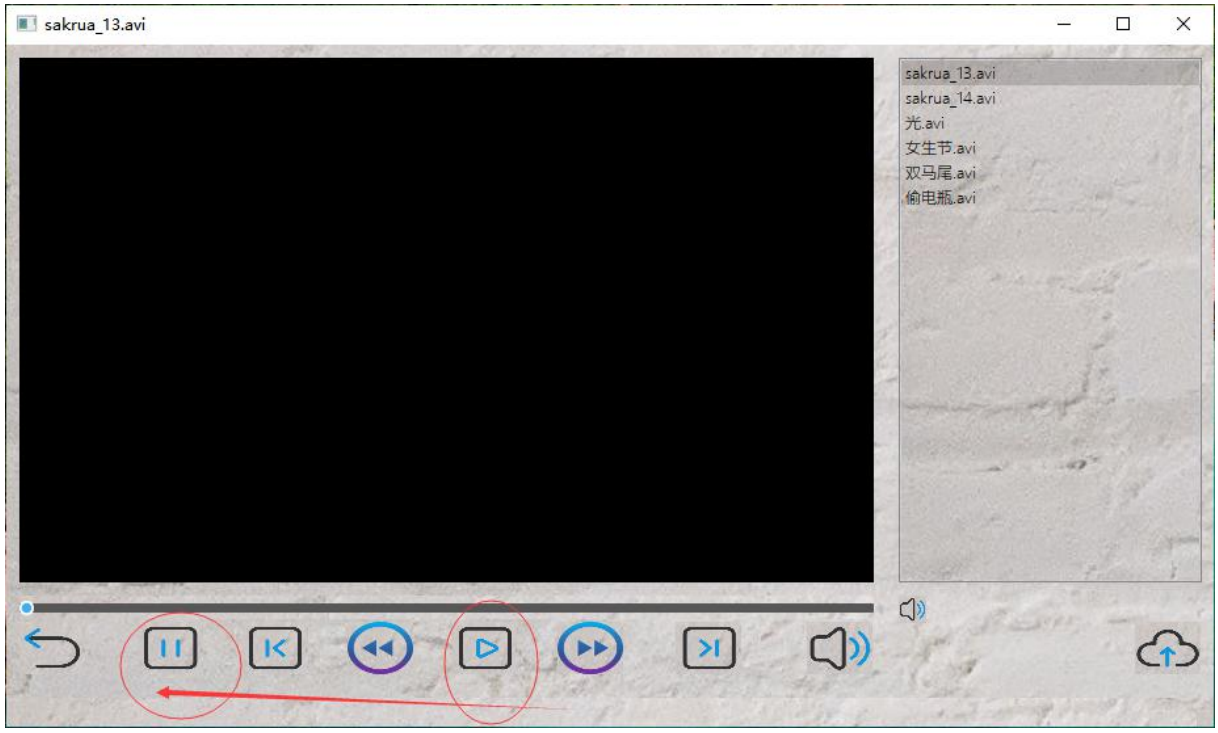



图 5.12

（4）按下上/下 会使切换视频。如图 5.13。


图 5.13

（5）按下快进和快退按钮会使视频前进或快退两秒。如图 5.14（快进两秒）。

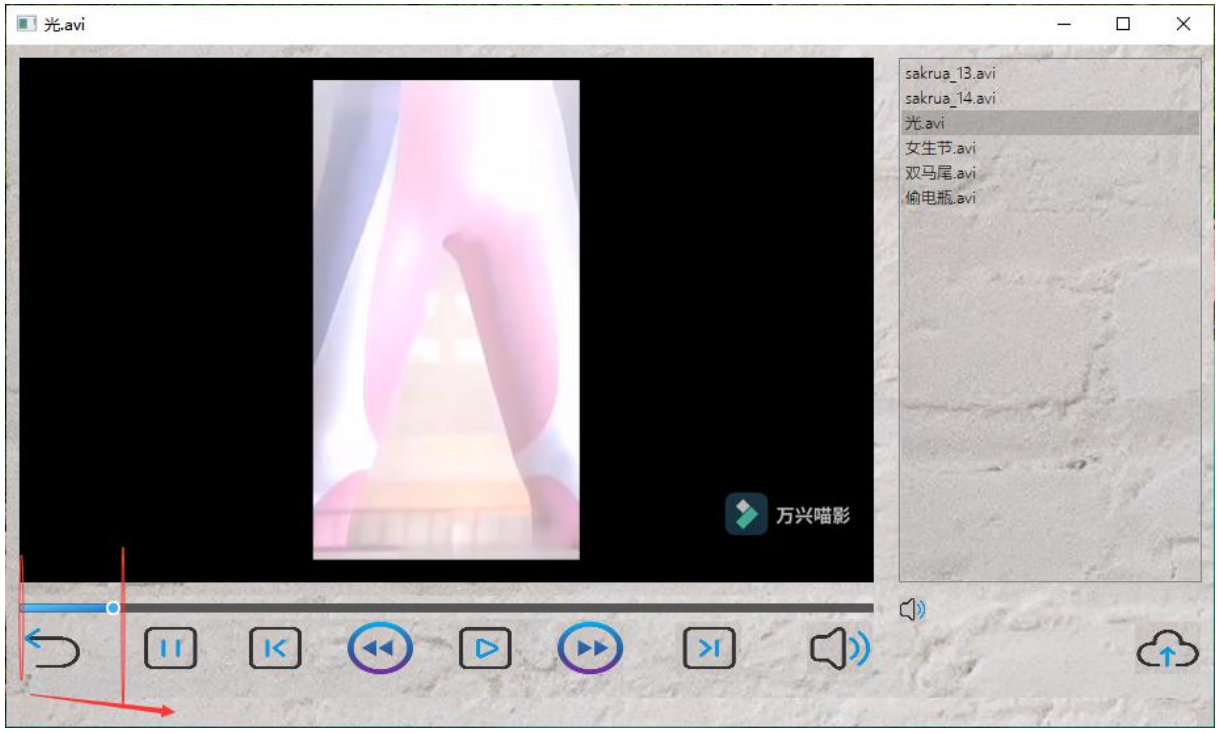



图 5.14

（6）按下静音，会禁掉声音；按下音量键会显示音量滑动条，通过滑动可调节音量的大小。 如图 5.15


图 5.15

## 5.3 天气查询测试

打开天气界面，上面显示长沙市今天的天气情况，下面部分显示未来六天内长沙市的天气情况。如图 5.16

图　5.16

## 5.4 地图查询测试

（1）通过鼠标或者界面左上角的按钮可完成对地图的缩放，如图 5.17,5.18 所示。

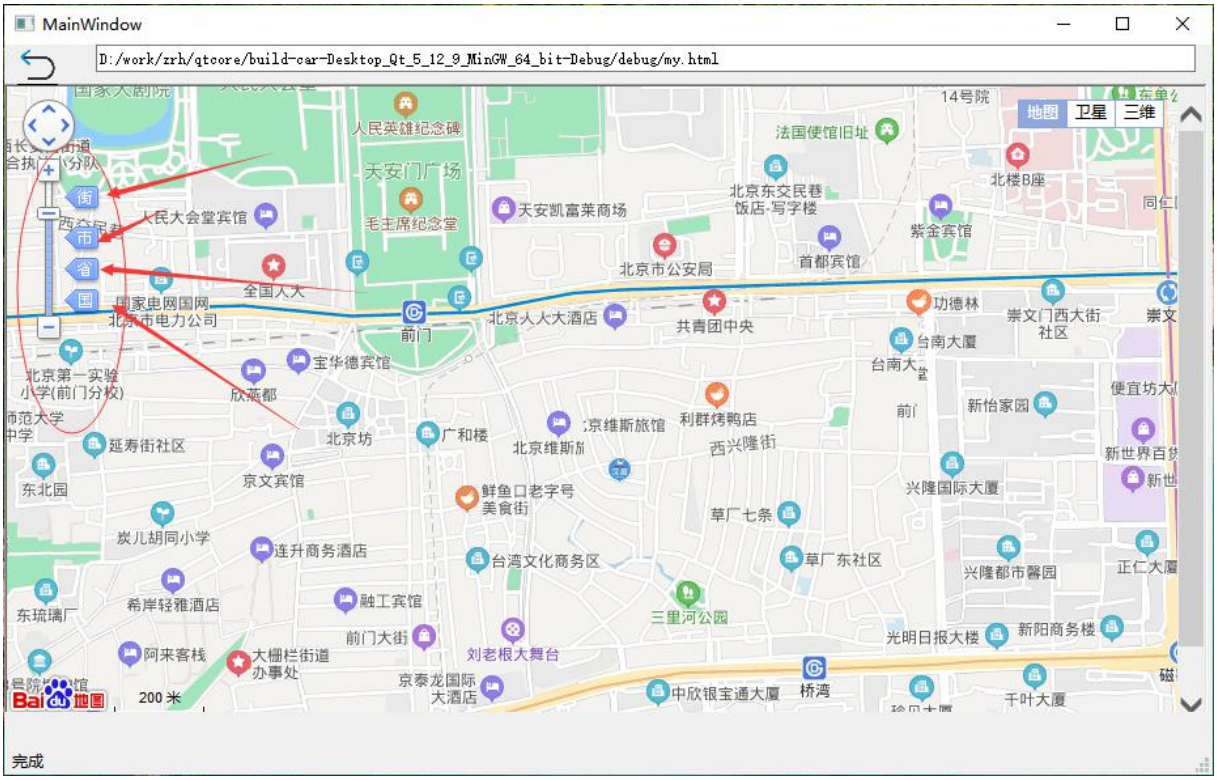



图　5.17


图　5.18

（2）通过鼠标或者地图左上角的控件可实现移动。如图 5.19 所示。

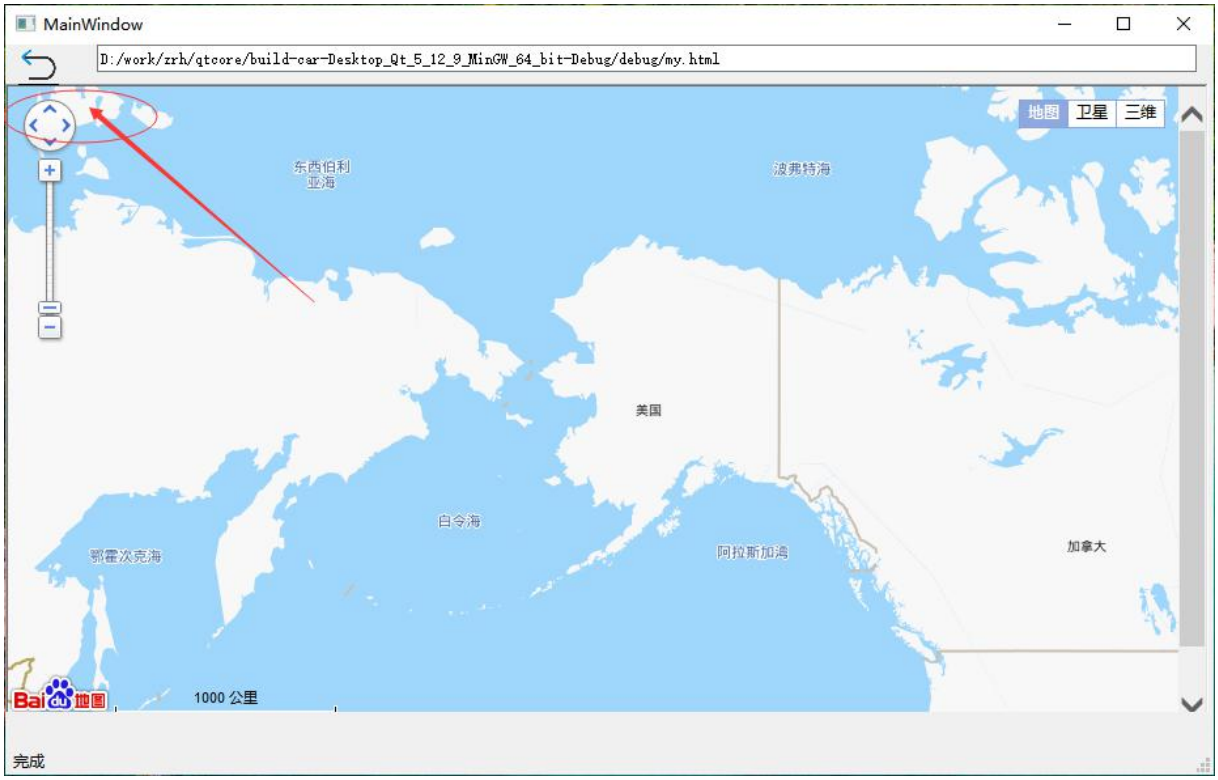



图 5.19

（3）通过点击右上角的按钮可切换不同的图层。如 5.20,5.21,5.22 所示。

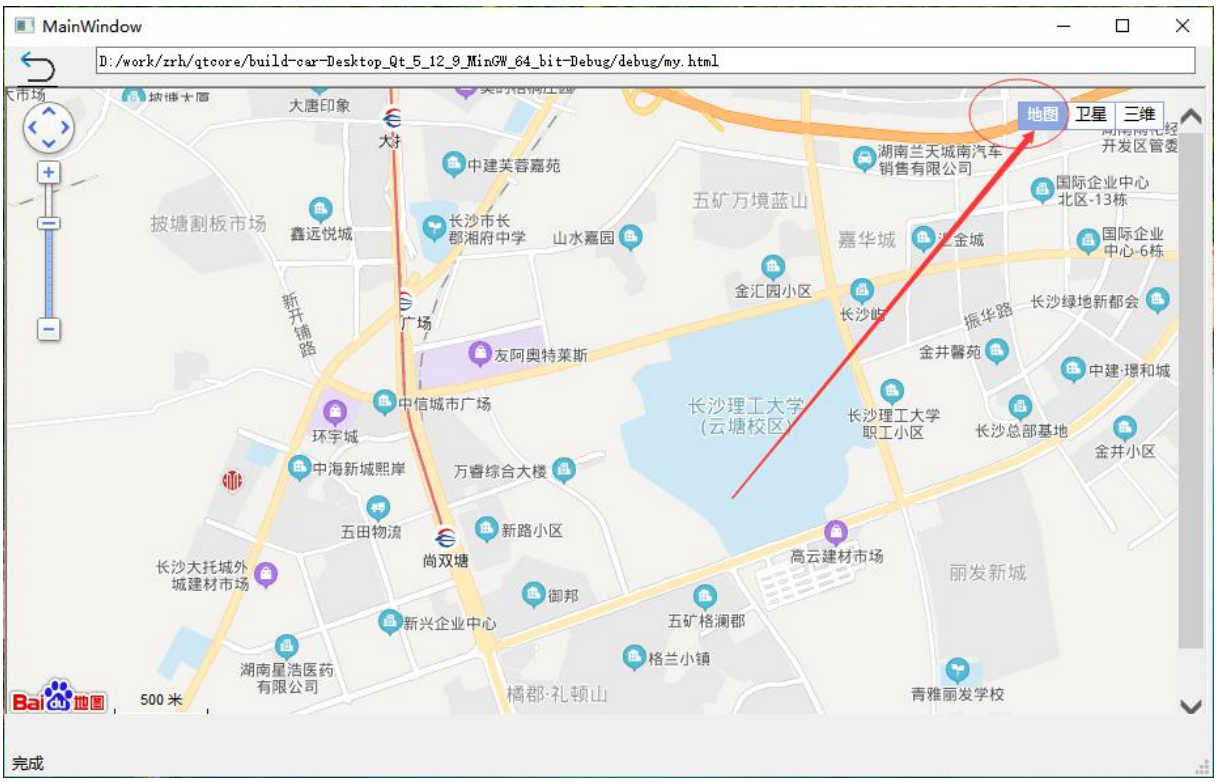



图 5.20

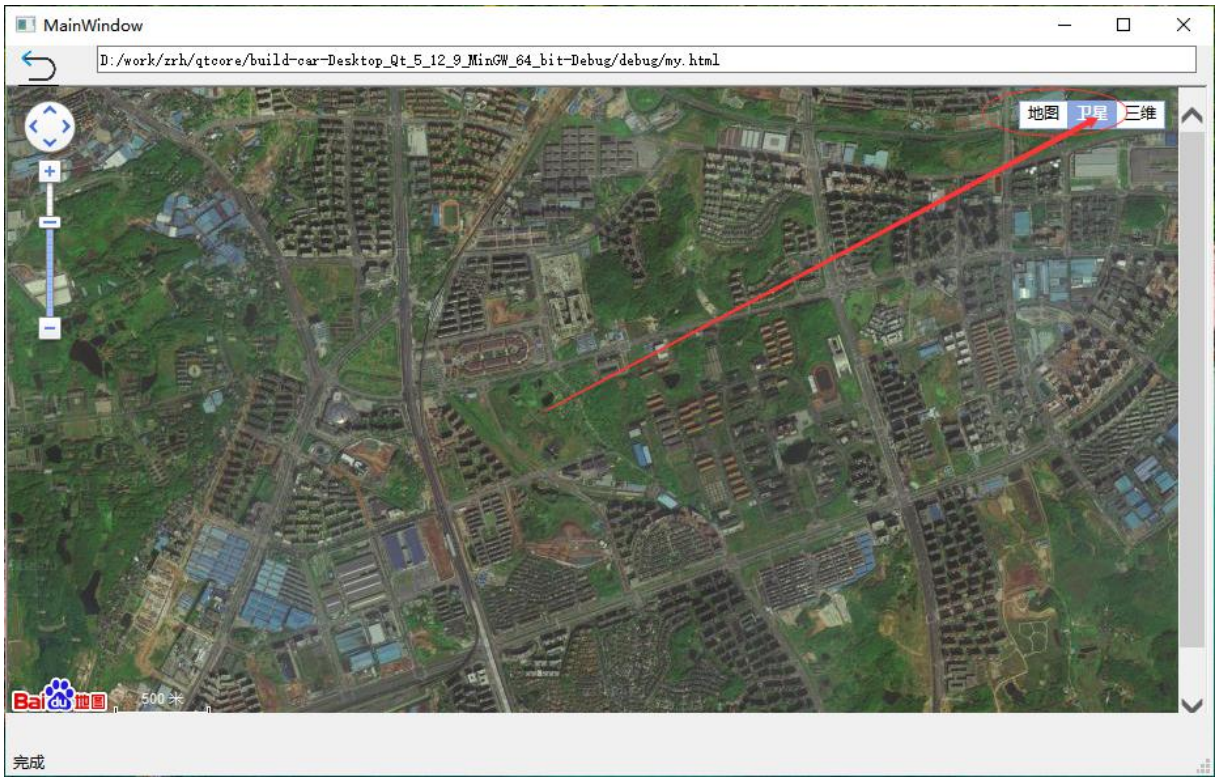

MainWindow
D:/work/zrh/qtcore/build-car-Desktop_Qt_5_12_9_MinGW_64_bit-Debug/debug/my.html
地图
卫星
三维
500 米
完成

图 5.21


MainWindow
D:/work/zrh/qtcore/build-car-Desktop_Qt_5_12_9_MinGW_64_bit-Debug/debug/my.html
地图
卫星
三维
显示卫星影像
披塘大厦
大唐印象
中建芙蓉嘉苑
湖南兰天城南汽车销售有限公司
五矿万境蓝山
披塘割板市场
鑫远悦城
长沙市长郡湘府中学
山水嘉园
嘉华城
汇金城
国际企业中心-6栋
金汇园小区
长沙屿
长沙绿地新都会
广场
友阿奥特莱斯
金井馨苑
中建·璟和城
中信城市广场
环宇城
长沙理工大学(云塘校区)
长沙理工大学职工小区
长沙总部基地
金井小区
中海新城熙岸
万睿综合大楼
五田物流
尚双塘
高云建材市场
丽发新城
湖南兴威建材大市场
御邦
新兴企业中心
五矿格澜郡
格兰小镇
湖南星浩医药有限公司
橘郡·礼顿山
青雅丽发学校
500 米
完成

图 5.22

# 6 总结

通过实训老师的课堂讲解与课后项目的实践，我对之前从未了解过的 Qt Creator、Keil5 等有了一定的了解，并且对 Qt Creator 的用法有了较深刻的认识，也在一定程度上培养了兴趣，并且累积了一定的专业知识。从而更加确定了自己以后的努力方向。要想在短短的不到三周的时间里完成实训任务，并且尽可能多的学到东西，就需要我们上课认真地听讲并且反复练习课上所学到的东西，加深对知识的了解，不懂的地方要及时询问老师或同学。只有我们跟老师和身边的同学多沟通，才能在真正意义上发挥实训的作用。

在这短短地不到三周的时间里，我深刻地感觉到自己在实训过程中知识与经验的缺乏，不少地方都遇到了问题，也让我真正体会到“学如逆水行舟，不进则退”的涵义。两位老师在课上所讲的知识全都是之前从未涉猎过的，也充满了陌生感，就像一切都从 0 开始，这也给了我莫大的动力与兴趣去学习。

自己做的东西还有很多欠缺的地方，希望将来有机会我可以不断去完善它，学习更多的知识，打下更牢固的技术。

# 7 参考文献

[1]白建平.QT 高级编程.[M].电子工业出版社.2010

[2]张思民.C++开发技术与工程实践[M].:清华大学出版社,2010.

[3]耿样义.C++实用教程[M]:机械工业出版社,2006.

# 8 附录:主要源代码

MainWindow.cpp //主界面窗口

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    //窗口分配空间
    music = new musicwindow(this);
    video = new videowindow(this);
    map = new mapwindow(this);
    weather = new weatherwindow(this);

    //显示时间
    getTime();
}

MainWindow::~MainWindow()
{
    delete ui;
}


//显示窗口
void MainWindow::on_music_bt_clicked()   //音乐
{
    music->show();
    this->hide();
}

void MainWindow::on_video_bt_clicked()
{
    video->show();
    this->hide();
}
```

```
void MainWindow::on_map_bt_clicked()
{
    map->show();
    this->hide();
}

void MainWindow::on_weather_bt_clicked()
{
    weather->show();
    this->hide();
}

void MainWindow::getTime()
{
    QDateTime time =QDateTime::currentDateTime();
    QString str= time.toString("yyyy-MM-dd hh:mm:ss dddd");
    ui->time->setText(str);
}
```

Mapwindow.cpp　//地图窗口

```
#include "mapwindow.h"
#include "ui_mapwindow.h"

mapwindow::mapwindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::mapwindow)
{
    ui->setupUi(this);

   // this->ui->axWidget->setControl("D:/work/zrh/qtcore/car/map.html");

    qApp->setApplicationName("padcollectionclient");
    QString sFilePath = QCoreApplication::applicationDirPath();
    //qDebug()<<"path:"<<sFilePath<<endl;
    sFilePath += "/my.html";

    ui->lineEdit->setText(sFilePath);
```

```
ui->axWidget->setControl(QString::fromUtf8("{8856F961-340A-11D0-A96B-00C04FD705A2}"));

    loadNavigate();
}

mapwindow::~mapwindow()
{
    delete ui;
}

void mapwindow::on_back_bt_clicked()
{
    this->parentWidget()->show();
    this->hide();
}

void mapwindow::loadNavigate()
{
    QString Utr = ui->lineEdit->text().trimmed();
    ui->axWidget->dynamicCall("Navigate(const QString&)",Utr);
}
```

Musicwindow.cpp

```
#include "musicwindow.h"
#include "ui_musicwindow.h"
#include <QDebug>

musicwindow::musicwindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::musicwindow)
{
    ui->setupUi(this);
    //读取文件
    readFile();
  //  on_playbtn_2_clicked();
    on_musicListRowNum(0);
    //初始化界面
    initUI();
    on_volume_clicked();
}
```

```
musicwindow::~musicwindow()
{
    delete ui;
}

void musicwindow::readFile()
{
    //设置遍历的目录
    str_pathmusic = "D:/work/zrh/qtcore/car/music";
    QDir dir(str_pathmusic);


    //设置文件过滤器
    QStringList nameFileters;
    //设置文件过滤器的格式
    nameFileters << "*.mp3";
    //将过滤后的文件名存入到 list 中
    QStringList files =
dir.entryList(nameFileters,QDir::Files|QDir::Readable,QDir::Name);
    //添加到 listwidget 中 item 中
    ui->list_music_2->addItems(files);
    //把 item 加入到 listwidget 中
    ui->list_music_2->sortItems();
    /*QMediaPlayer * player = new QMediaPlayer;

player->setMedia(QUrl::fromLocalFile("D:\\work\\zrh\\qtcore\\car\\music\\4
円 (ヨエン) - アイロニ (反语).mp3"));
    player->setVolume(100);
    player->play();*/

    p_item = ui->list_music_2->item(0);
    //qDebug()<<"music name:"<<p_item->text();
    p_item->setSelected(true);
    // p_player = new QMediaPlayer(this);

connect(ui->list_music_2,SIGNAL(currentRowChanged(int)),this,SLOT(on_musicL
istRowNum(int)));
}
void musicwindow::on_musicListRowNum(int n_number)
{
   // qDebug()<<n_number;
   // qDebug()<<ui->list_music_2->item(n_number)->text()<<endl;
    str_music = ui->list_music_2->item(n_number)->text();
    p_player = new QMediaPlayer(this);
```

```
    p_player->setMedia(QUrl::fromLocalFile(str_pathmusic +'/'+str_music));
    //qDebug()<<str_music<<" "<<str_pathmusic<<endl;
    p_player->setVolume(100);
    //p_player->stop();
    n_music_number = n_number;
   // qDebug()<<"b:"<<b_play_music<<endl;
}



void musicwindow::on_back_bt_clicked()
{
    this->parentWidget()->show();
    this->hide();
}
void musicwindow::initUI()
{

}


void musicwindow::on_playbtn_2_clicked()
{
        //qDebug()<<"b_play           _music = " << b_play_music <<endl;
        if(b_play_music)
        {


ui->playbtn_2->setStyleSheet(QString("QPushButton#playbtn_2{border-image:
url(:/new/prefix1/images/play2.png); background-color: transparent;}"));
    //      ui->playbtn->setStyleSheet(QString("border-image:
url(:/new/prefix1/pause2.png);"));
            b_play_music = false;
            p_player->pause();

        }
        else
        {

ui->playbtn_2->setStyleSheet(QString("QPushButton#playbtn_2{border-image:
url(:/new/prefix1/images/pause2.png); background-color: transparent;}"));
    //      ui->playbtn->setStyleSheet(QString("border-image:
url(:/new/prefix1/play2.png);"));
            b_play_music = true;
            p_player->play();
```

```
            // qDebug()<<"b_play          _music = " << b_play_music <<endl;

        }
}

void musicwindow::on_upbtn_2_clicked()
{
    // p_player->stop();
     if(n_music_number == 0)
     {
         int nMax = ui->list_music_2->count();
         //qDebug()<<nMax;
         str_music = ui->list_music_2->item(nMax - 1)->text();
         p_item = ui->list_music_2->item(nMax - 1);
         p_item->setSelected(true);

         QString str = str_pathmusic + '/' +str_music;
         p_player->setMedia(QUrl::fromLocalFile(str));
         p_player->setVolume(100);
         p_player->stop();
         n_music_number = nMax - 1;

     }
     else
     {

         str_music = ui->list_music_2->item(n_music_number - 1)->text();
         p_item = ui->list_music_2->item(n_music_number - 1);
         p_item->setSelected(true);

         QString str = str_pathmusic + '/' +str_music;
         p_player->setMedia(QUrl::fromLocalFile(str));
         p_player->setVolume(100);
         p_player->stop();
         n_music_number -= 1;

     }


}

void musicwindow::on_down_btn_clicked()
{
    int nMax = ui->list_music_2->count();
    if(n_music_number == (nMax - 1))
```

```
    {
        str_music = ui->list_music_2->item(0)->text();
        p_item = ui->list_music_2->item(0);
        p_item->setSelected(true);

        QString str = str_pathmusic + '/' +str_music;
        p_player->setMedia(QUrl::fromLocalFile(str));
        p_player->setVolume(100);
        p_player->stop();
        n_music_number = 0;
    }
    else
    {
        str_music = ui->list_music_2->item(n_music_number + 1)->text();
        p_item = ui->list_music_2->item(n_music_number + 1);
        p_item->setSelected(true);

        QString str = str_pathmusic + '/' +str_music;
        p_player->setMedia(QUrl::fromLocalFile(str));
        p_player->setVolume(100);
        p_player->stop();
        n_music_number += 1;
    }

}


void musicwindow::on_volume_clicked()
{
    if(m_checked)
    {
        ui->vol_change->show();
        m_checked = false;//点击显示
        //qDebug()<<"check :"<<m_checked<<endl;
    }
    else
    {
        ui->vol_change->hide();
        m_checked = true;
       // qDebug()<<"check :"<<m_checked<<endl;
    }
}

void musicwindow::on_vol_change_sliderMoved()
{
```

```
    p_player->setVolume(ui->vol_change->value());
}

/*void musicwindow::on_toolButton_5_clicked()
{
    QStringList
fileList=QFileDialog::getOpenFileNames(this,QString::fromLocal8Bit("打开文
件"),"D:/",tr("*.mp3"));

   if (fileList.count()<1)
        return;

    for (int i=0; i<fileList.count();i++)
    {
       QString aFile=fileList.at(i);
       playlist->addMedia(QUrl::fromLocalFile(aFile));  //添加文件
       QFileInfo fileInfo(aFile);
       ui->list_music_2->addItem(fileInfo.fileName()); //添加到界面文件列表
    }
   if(p_player->state() != QMediaPlayer::PlayingState)
        playlist->setCurrentIndex(0);
    p_player->play();
}*/
```

Videowindow.cpp  //视频窗口

```
#include "videowindow.h"
#include "ui_videowindow.h"
#include<QMediaPlayer>
#include<QUrl>
#include<QFileDialog>
#include <QDialog>
#include<QDateTime>
#include<QPixmap>
#include<QPalette>
#include<QBrush>
#include<QWidget>
#include<QListWidgetItem>

videowindow::videowindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::videowindow)
```

```
{
    ui->setupUi(this);
    on_volume_clicked();
    //readFile();


    mediaplayer = new QMediaPlayer(this);
    playlist = new QMediaPlaylist(this);
    playlist->setPlaybackMode(QMediaPlaylist::Loop);
    mediaplayer->setVideoOutput(ui->widget);
    mediaplayer->setPlaylist(playlist);
    //设置播放属性
    //监听

connect(mediaplayer,SIGNAL(stateChanged(QMediaPlayer::State)),this,SLOT(med
iaStateChange(QMediaPlayer::State)));
      //播放进度函数

connect(mediaplayer,SIGNAL(positionChanged(qint64)),this,SLOT(positionChang
ed(qint64)));

connect(mediaplayer,SIGNAL(durationChanged(qint64)),this,SLOT(durationChang
ed(qint64)));

connect(mediaplayer,SIGNAL(positionChanged1(qint64)),this,SLOT(positionChan
ged1(qint64)));

//connect(mediaplayer,SIGNAL(error(QMediaPlayer::Error)),this,SLOT(handleEr
ror()));
    connect(playlist,SIGNAL(currentIndexChanged(int)), this,
SLOT(onPlaylistChanged(int)));

}

videowindow::~videowindow()
{
    delete ui;
}

void videowindow::on_back_bt_clicked()   // 返回主界面
{
    this->parentWidget()->show();
    this->hide();
}
```

```
void videowindow::play(){              //获取播放状态

    switch (mediaplayer->state()) {
    case QMediaPlayer::PlayingState:
        mediaplayer->pause();
        break;
    default:
         mediaplayer->play();
    break;
}
}
void videowindow::onPlaylistChanged(int position) //
{
    ui->listWidget->setCurrentRow(position);
    QListWidgetItem  *item=ui->listWidget->currentItem();
    if (item)
        this->setWindowTitle(item->text());
}


void videowindow::setPosition(int position)     //获取进度条位置
{

    mediaplayer->setPosition(position);

}
void videowindow::durationChanged(qint64 duration)
{

    ui->horizontalSlider->setRange(0,duration);
}
 void videowindow::positionChanged(qint64 position)    //改变播放位置
 {

     ui->horizontalSlider->setValue(position);
 }



void videowindow::on_horizontalSlider_sliderMoved(int position)
{
    setPosition(position);
}
```

```
void videowindow::on_toolButton_clicked()     //播放
{
   play();
}


void videowindow::mediaStateChange(QMediaPlayer::State state)     //播放过程中调整动态函数
{

    switch (state) {
    case QMediaPlayer::PlayingState:
      ui->toolButton->setToolTip("暂停");
     // ui->toolButton->setIcon(QPixmap(":/new/prefix1/2.png"));
        break;
    default:
        ui->toolButton->setToolTip("播放");
       // ui->toolButton->setIcon(QPixmap(":/new/prefix1/1.png"));
        break;
    }
}

void videowindow::on_toolButton_5_clicked()    //打开文件
{
       QStringList fileList=QFileDialog::getOpenFileNames(this,QString::fromLocal8Bit("打开文件"),"D:/work/zrh/qtcore/car/video",tr("*.wmv *.mp3 *.avi"));

       if (fileList.count()<1)
            return;

       for (int i = 0; i<fileList.count();i++)
        {
           QString aFile=fileList.at(i);
           playlist->addMedia(QUrl::fromLocalFile(aFile));  //添加文件
           QFileInfo fileInfo(aFile);
           ui->listWidget->addItem(fileInfo.fileName()); //添加到界面文件列表
        }

       if (mediaplayer->state() != QMediaPlayer::PlayingState)
          playlist->setCurrentIndex(0);

       mediaplayer->play();
```

```
}


void videowindow::on_listWidget_doubleClicked(const QModelIndex &index)   //
双击文件名播放
{
        int rowNo=index.row();
        playlist->setCurrentIndex(rowNo);
        mediaplayer->play();

}


void videowindow::on_toolButton_8_clicked()  //上一个
{
    playlist->previous();

}

void videowindow::on_toolButton_9_clicked()  // down
{
    playlist->next();

}




void videowindow::on_toolButton_10_clicked()   //快进
{
   qint64 t;
   t = mediaplayer->position();
   t += 2000;
   mediaplayer->setPosition(t);
}




void videowindow::on_toolButton_11_clicked()  //快退
{
    qint64 t;
    t = mediaplayer->position();
    t -= 2000;
```

```
    mediaplayer->setPosition(t);
}



void videowindow::on_toolButton_4_clicked()    //静音
{
    bool mute=mediaplayer->isMuted();
    mediaplayer->setMuted(!mute);

   /* if (mute)
        ui->toolButton_4->setText("开启静音");
    else
        ui->toolButton_4->setText("关闭静音");*/
}


void videowindow::on_toolButton_3_clicked()
{
   mediaplayer->stop();
}


//void videowindow::readFile()
//{
//    path_video = "D:/work/zrh/qtcore/car/video";
//    QDir dir(path_video);

//    QStringList nameFileters;
//    nameFileters << "*.avi";
//    QStringList files =
dir.entryList(nameFileters,QDir::Files|QDir::Readable,QDir::Name);

//    ui->listWidget->addItems(files);

//    ui->listWidget->sortItems();
//    qDebug()<<"我进来啦"<<endl;

//   // playlist = ui->listWidget->item(0);
//    //playlist->setCurrentIndex(0);
//}
```

```
void videowindow::on_volume_clicked() //声音按钮显示
{
    if(m_checked)
    {
        ui->vol_change->show();
        m_checked = false;//点击显示
        //qDebug()<<"check :"<<m_checked<<endl;
    }
    else
    {
        ui->vol_change->hide();
        m_checked = true;
       // qDebug()<<"check :"<<m_checked<<endl;
    }
}
```

```
void videowindow::on_vol_change_sliderMoved()
{
    mediaplayer->setVolume(ui->vol_change->value());
}
```

**Weatherwindow.cpp　　//天气窗口**

```
#include "weatherwindow.h"
#include "ui_weatherwindow.h"

weatherwindow::weatherwindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::weatherwindow)
{
    ui->setupUi(this);

    manager = new QNetworkAccessManager(this);
    //关联槽函数
    connect(manager, SIGNAL(finished(QNetworkReply *)),this,
SLOT(replyFinished(QNetworkReply *)));

manager->get(QNetworkRequest(QUrl("http://t.weather.itboy.net/api/weather/c
ity/101250101")));
    //http://t.weather.itboy.net/api/weather/city/101250101
    //http://t.weather.itboy.net/api/weather/city/101280101
```

```
    //对 qlabel 进行赋值
    dw[0]=ui->dateweekday;
    dw[1]=ui->dateweekday_2;
    dw[2]=ui->dateweekday_3;
    dw[3]=ui->dateweekday_4;
    dw[4]=ui->dateweekday_5;
    dw[5]=ui->dateweekday_6;

    tpe1[0]=ui->type1;
    tpe1[1]=ui->type1_2;
    tpe1[2]=ui->type1_3;
    tpe1[3]=ui->type1_4;
    tpe1[4]=ui->type1_5;
    tpe1[5]=ui->type1_6;

    hg[0]=ui->high;
    hg[1]=ui->high_2;
    hg[2]=ui->high_3;
    hg[3]=ui->high_4;
    hg[4]=ui->high_5;
    hg[5]=ui->high_6;

    lw[0]=ui->low;
    lw[1]=ui->low_2;
    lw[2]=ui->low_3;
    lw[3]=ui->low_4;
    lw[4]=ui->low_5;
    lw[5]=ui->low_6;
}

weatherwindow::~weatherwindow()
{
    delete ui;
}

void weatherwindow::on_back_bt_clicked()
{
    this->parentWidget()->show();
    this->hide();
}


void weatherwindow::replyFinished(QNetworkReply *arg)//对返回的数据进行 json
解析处理
{
```

```
QString js = arg->readAll();
QJsonParseError error;
QJsonDocument json = QJsonDocument::fromJson(js.toUtf8(),&error);
if(error.error != QJsonParseError::NoError)
{
    qDebug()<<error.errorString();
}
else
    qDebug()<<"init json succeed!";
//取对象
QJsonObject obj=json.object();

//对象里的 data
QJsonValue value1 = obj.take("data");
QJsonObject obj1=value1.toObject();

QJsonValue value2 = obj.take("cityInfo");
QJsonObject obj2=value2.toObject();

//cityInfo 里的 city
QJsonValue value_city = obj2.take("city");
//qDebug()<<value_city.toString();
ui->city->setText(value_city.toString());

//updatetime
QJsonValue value_upt = obj2.take("updateTime");
//qDebug()<<value_upt.toString();
 ui->updatetime->setText("更新时间 "+value_upt.toString());

//centigrade
QJsonValue value_centg = obj1.take("wendu");
//qDebug()<<value_centg.toString();
ui->centigrade->setText(value_centg.toString()+"℃");

//apilevel
QJsonValue value_apilevel = obj1.take("quality");
//qDebug()<<value_apilevel.toString();
ui->aqilevel->setText("级别 "+value_apilevel.toString());

//humid
QJsonValue value_humid = obj1.take("shidu");
//qDebug()<<value_humid.toString();
ui->humid->setText("湿度 "+value_humid.toString());

//data 里的 forecast
```

```
    QJsonValue value = obj1.take("forecast");
    QJsonArray arr = value.toArray();

     QVariantList list = arr.toVariantList();
     QVariantMap map = list[0].toMap();
     ui->weekday->setText(map["week"].toString());
     ui->date_2->setText(map["ymd"].toString());

     ui->aqi->setText("空气质量指数 "+QString::number(map["aqi"].toInt()));
     ui->wind->setText("风速 "+map["fx"].toString() +" "+
map["fl"].toString());
     ui->type->setText(map["type"].toString());
     if(map["type"].toString()=="晴")

ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/sunny.p
ng")));
     if(map["type"].toString()=="多云")

ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/cloudy.
png")));
     if(map["type"].toString()=="阴")

ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/yintian.
png")));
     if(map["type"].toString()=="小雨")

ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/xiaoyu.
png")));
     if(map["type"].toString()=="霾")

ui->icon->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/mai.png
")));

     for(int i=0;i<6;i++)
     {
         QVariantMap map = list[i].toMap();

dw[i]->setText(map["date"].toString()+"("+map["week"].toString()+")");
         hg[i]->setText(map["high"].toString().mid(3).remove("℃")+"° ");
         lw[i]->setText(map["low"].toString().mid(3).remove("℃")+"° ");
         if(map["type"].toString()=="晴")
         {

tpe1[i]->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/sunny.pn
g")));
```

```
        }
        if(map["type"].toString()=="多云")
        {

tpe1[i]->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/cloudy.p
ng")));
        }
        if(map["type"].toString()=="阴")
        {

tpe1[i]->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/yintian.
png")));
        }
        if(map["type"].toString()=="小雨")
        {

tpe1[i]->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/xiaoyu.p
ng")));
        }
        if(map["type"].toString()=="霾")
        {

tpe1[i]->setPixmap(QPixmap(QString::fromUtf8(":/new/prefix1/images/mai.png")
));
        }
    }
}
```