# 1. 开发工程准备

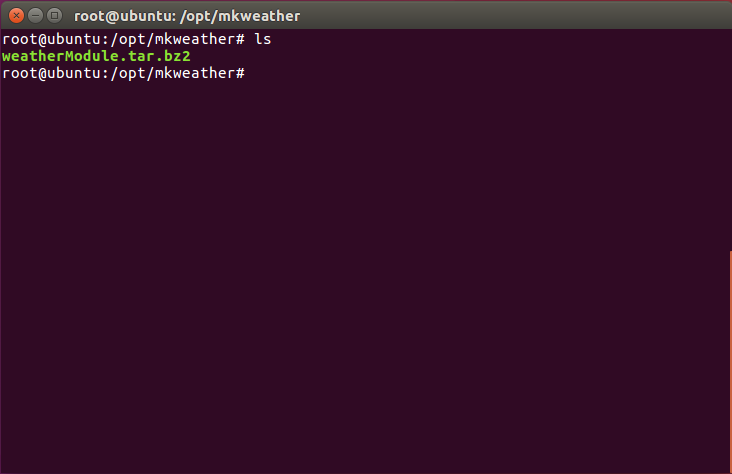
## 1.1 工程源码准备

\*工程目录准备

mkdir /opt/mkweather

cd /opt/mkweather

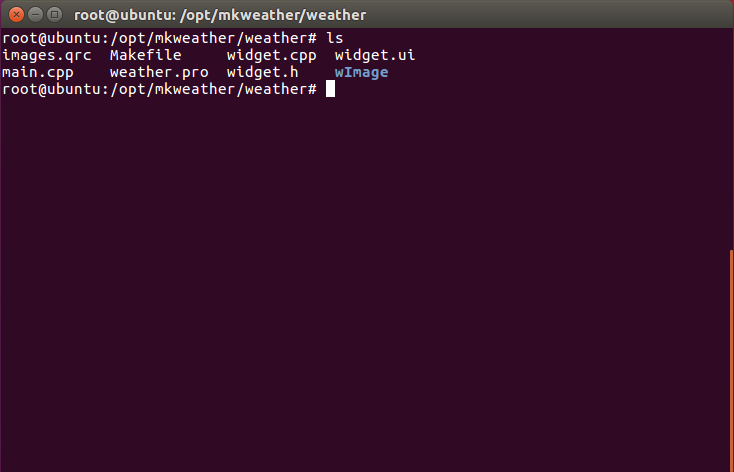
\*将demo工程包weatherModule.tar.bz2共享放置到此目录下



\*解压进入

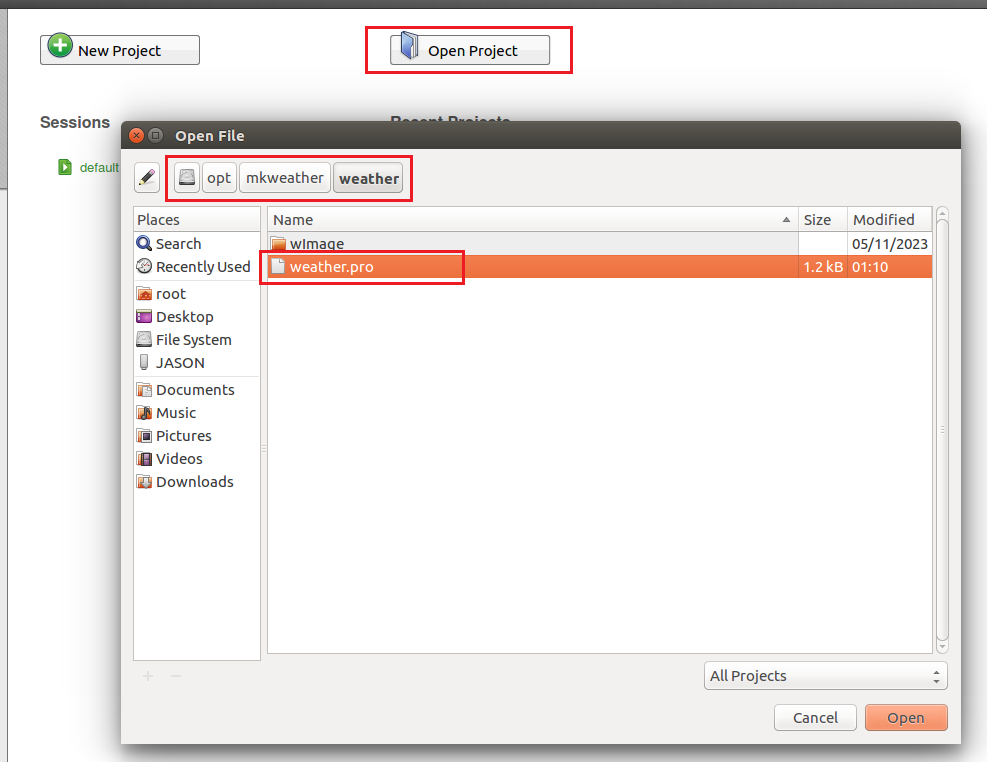
tar xvfj weatherModule.tar.bz2

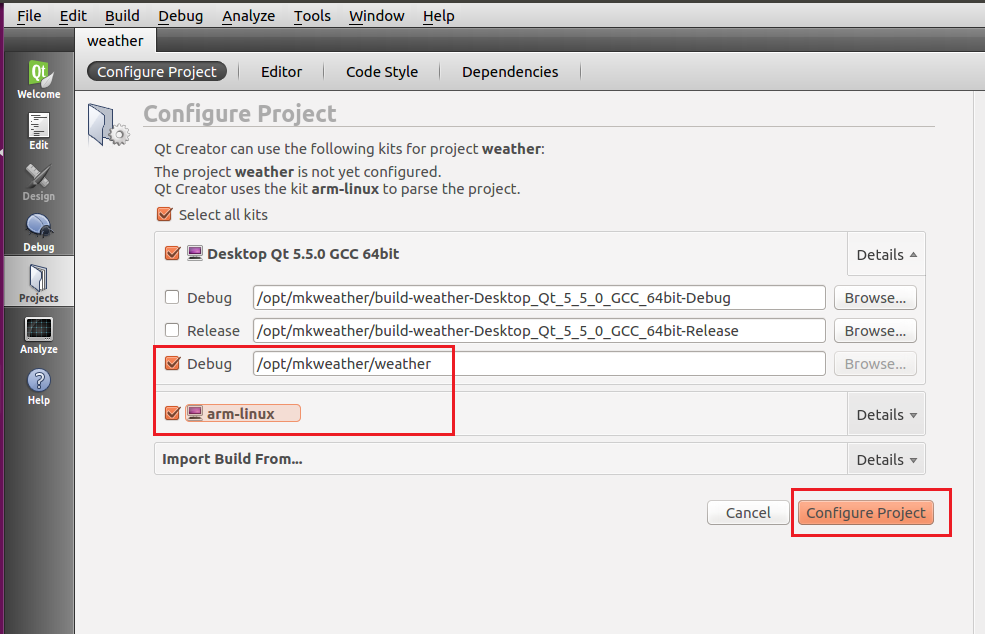
cd weather



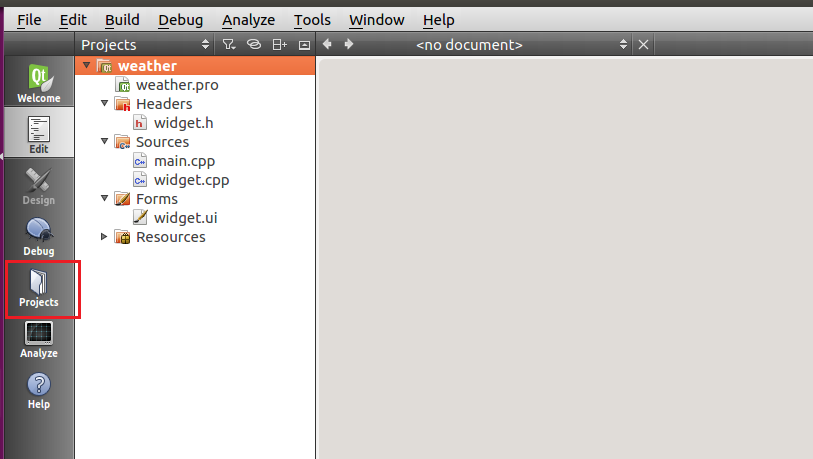
## 1.2 QT工程加载

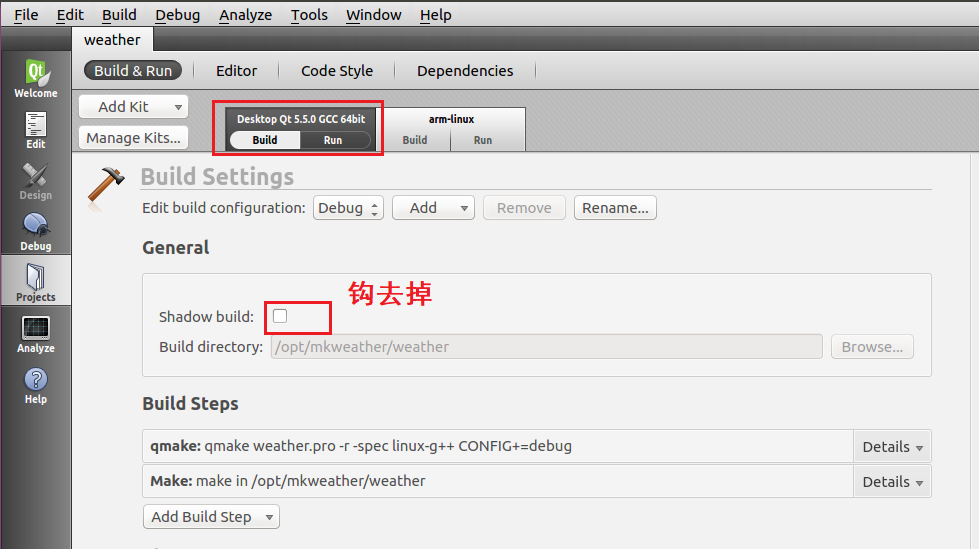
\*打开qtcreator，加载上面工程源码包



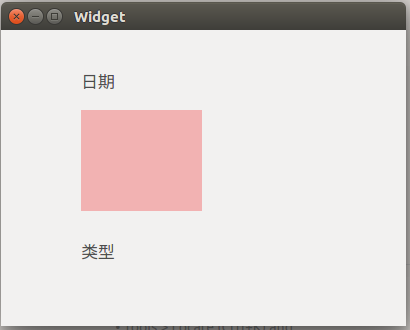


\*工程配置





\*点击左下角编译执行按键执行，可以看到



# 2. 天气数据获取

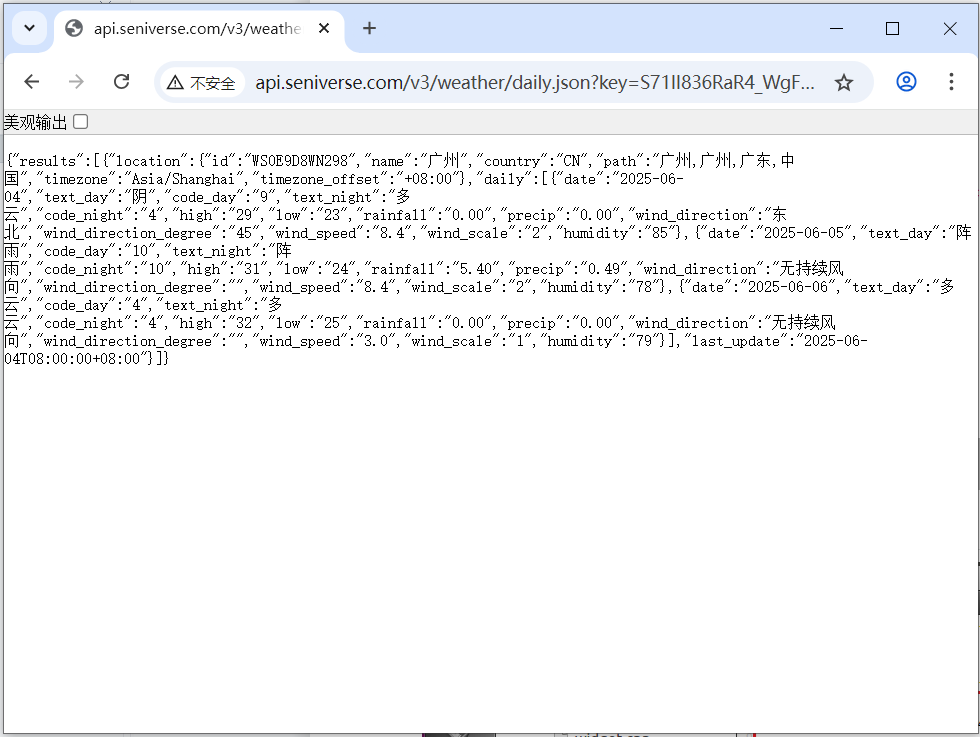
## 2.1 天气API测试

**检测API接口是否有效**

天气API接口：

http://api.seniverse.com/v3/weather/daily.json?key=S71II836RaR4\_WgF2&location=guangzhou&language=zh-Hans&unit=c&start=-1&days=5"

将上述API接口输入到浏览器中，出现以下现象：



证明该API接口可以使用

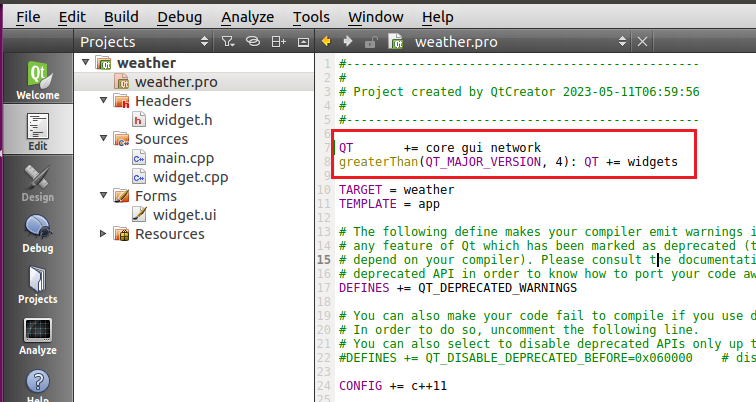
## 2.2 程序开发测试

双击weather.pro文件中，添加network

QT += core gui network

greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets

如下图示：



双击打开widget.h文件，添加以下同内容：

#include <QNetworkAccessManager>

#include <QNetworkReply>

#include <QNetworkRequest>

#include <QUrl>

namespace Ui {

class Widget;

}

class Widget : public QWidget

{

Q\_OBJECT

public:

explicit Widget(QWidget \*parent = nullptr);

~Widget();

void mSleep(unsigned int msec);

private slots:

void update\_weather();

void read\_weather(QNetworkReply\*reply);

void show\_weatherData();

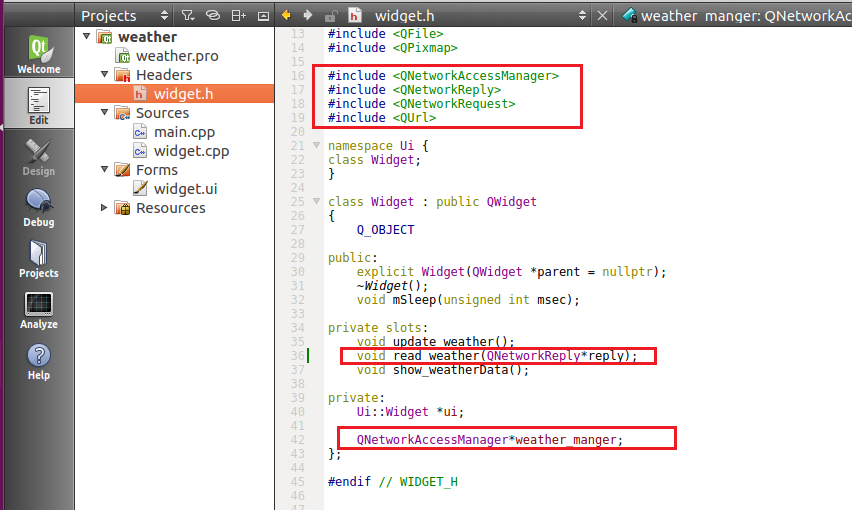
private:

Ui::Widget \*ui;

QNetworkAccessManager\*weather\_manger;

};

如下图示：



在widget.cpp文件的Widget函数中，添加：

Widget::Widget(QWidget \*parent) :

QWidget(parent),

ui(new Ui::Widget)

{

ui->setupUi(this);

weather\_manger = new QNetworkAccessManager();

connect(weather\_manger,SIGNAL(finished(QNetworkReply\*)),this,SLOT(read\_weather(QNetworkReply\*)));

update\_weather();

}

如下图示：



在widget.cpp文件的update\_weather函数中，添加：

//http请求天气数据

void Widget::update\_weather()

{

QUrl url;

url.setUrl("http://api.seniverse.com/v3/weather/daily.json?key=S71II836RaR4\_WgF2&location=guangzhou&language=zh-Hans&unit=c&start=-1&days=5");

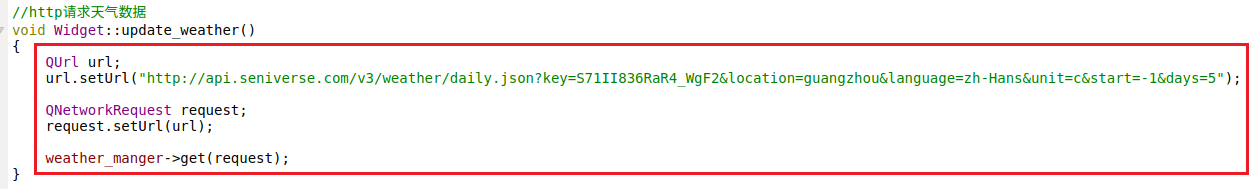
QNetworkRequest request;

request.setUrl(url);

weather\_manger->get(request);

}

如下图所示：



在menu.cpp文件的read\_weather函数中，添加：

//读取解析天气数据

void Widget::read\_weather(QNetworkReply\*reply)

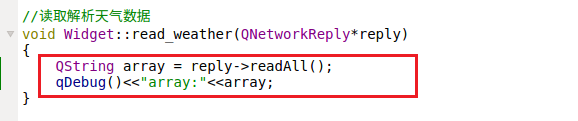
{

QString array = reply->readAll();

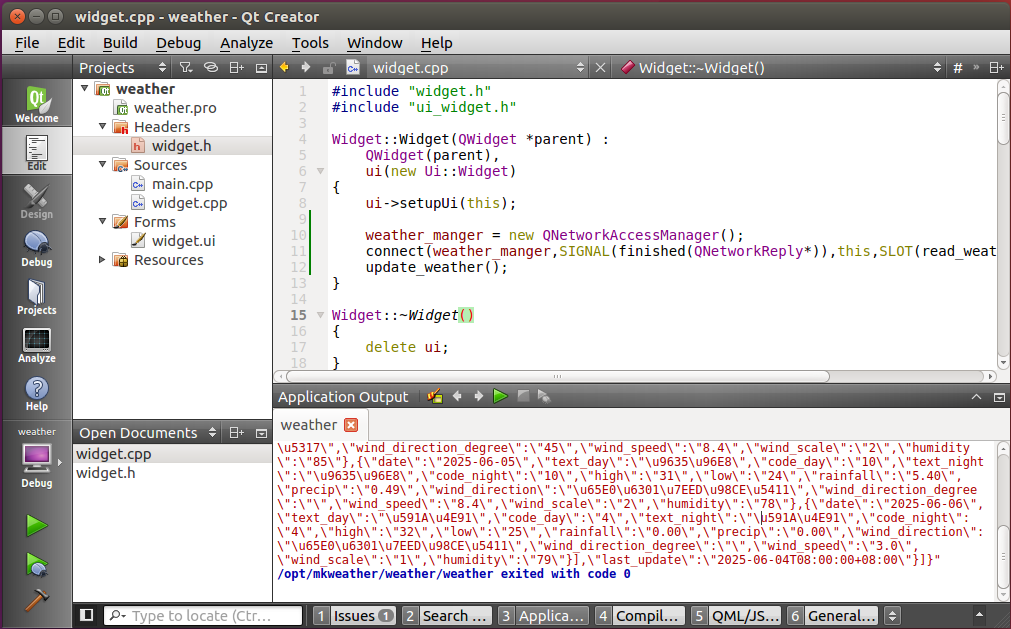
qDebug()<<"array:"<<array;

}

如下图所示：



点击左下角调试按键调试，最终效果演示：



说明可以正常获取到天气数据（JSON格式）

# 3. 天气数据解析

双击打开widget.h文件，添加以下同内容：

#include <QByteArray>

#include <QJsonParseError>

#include <QJsonArray>

#include <QJsonObject>

#include <QJsonDocument>

class WeatherData

{

public:

QString date;

QString type;

QString temp\_h;

QString temp\_l;

QString wind\_s;

QString wind\_d;

};

namespace Ui {

class Widget;

}

class Widget : public QWidget

{

Q\_OBJECT

public:

explicit Widget(QWidget \*parent = nullptr);

~Widget();

void mSleep(unsigned int msec);

private slots:

void update\_weather();

void read\_weather(QNetworkReply\*reply);

void show\_weatherData();

private:

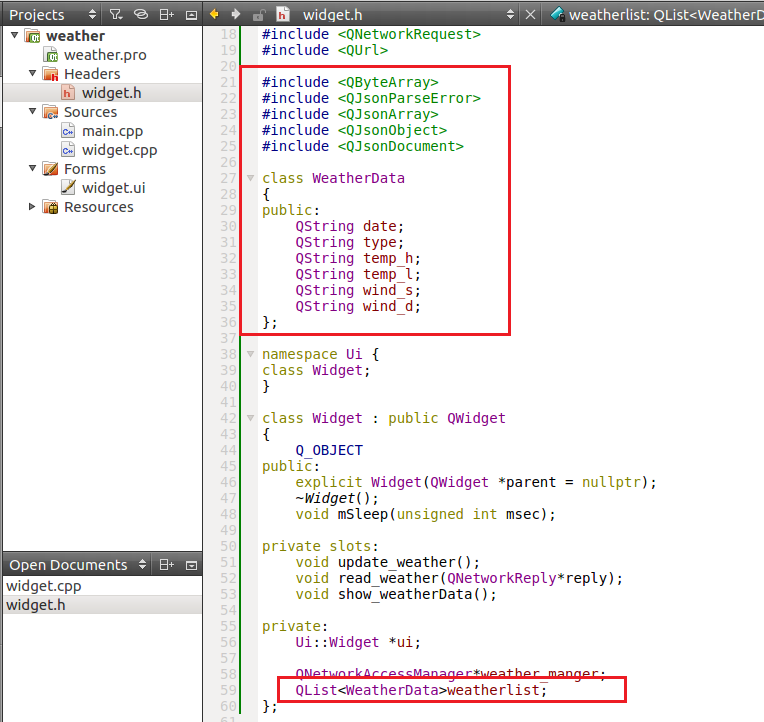
Ui::Widget \*ui;

QNetworkAccessManager\*weather\_manger;

QList<WeatherData>weatherlist;

};

如下图所示：



在widget.cpp文件的read\_weather函数中，添加：

//读取解析天气数据

void Widget::read\_weather(QNetworkReply\*reply)

{

QString array = reply->readAll();

//qDebug()<<"array:"<<array;

QJsonParseError err;

QJsonDocument Doc = QJsonDocument::fromJson(array.toUtf8(),&err);

if(err.error != QJsonParseError::NoError)

{

qDebug()<<"error";

return;

}

QJsonObject obj= Doc.object();

QJsonArray arr;

arr = obj.value("results").toArray();

obj = arr.at(0).toObject();

arr = obj.value("daily").toArray();

for(int i=0; i<arr.size(); i++)

{

obj = arr.at(i).toObject();

WeatherData wdata;

wdata.date = obj.value("date").toString();

wdata.type = obj.value("text\_day").toString();

wdata.temp\_l = obj.value("low").toString();

wdata.temp\_h = obj.value("high").toString();

wdata.wind\_d = obj.value("wind\_direction").toString();

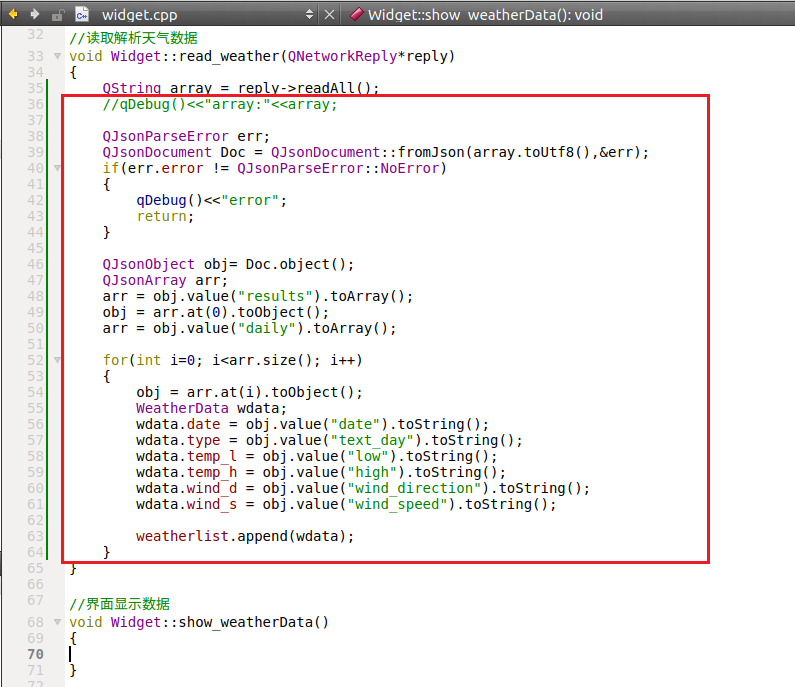
wdata.wind\_s = obj.value("wind\_speed").toString();

weatherlist.append(wdata);

}

}

如下图所示：



在widget.cpp的widget函数中，添加：

Widget::Widget(QWidget \*parent) :

QWidget(parent),

ui(new Ui::Widget)

{

ui->setupUi(this);

weather\_manger = new QNetworkAccessManager();

connect(weather\_manger,SIGNAL(finished(QNetworkReply\*)),this,SLOT(read\_weather(QNetworkReply\*)));

update\_weather();

QTimer::singleShot(1000,this,SLOT(show\_weaterData()));

}

如下图所示：



在widget.cpp中新建show\_weatherData函数：

//界面显示数据

void Widget::show\_weatherData()

{

if(weatherlist.isEmpty())

return;

QList<WeatherData> list = weatherlist;

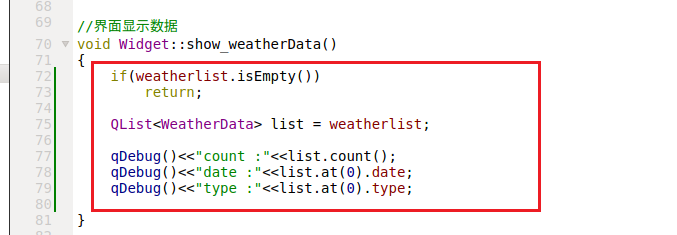
qDebug()<<"count :"<<list.count();

qDebug()<<"date :"<<list.at(0).date;

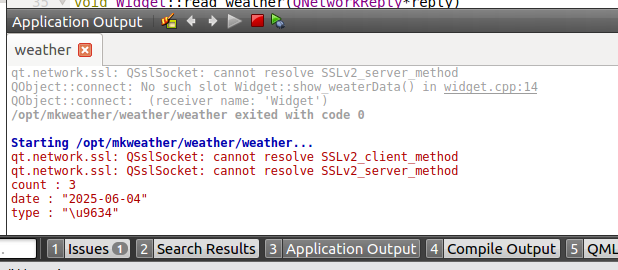
qDebug()<<"type :"<<list.at(0).type;

}

如下图所示：



点击编译调试，效果演示



可以看到正常解析出了天气的数据。

# 4. 天气数据显示

在widget.cpp的show\_weaterData函数中，添加：

//界面显示数据

void Widget::show\_weatherData()

{

if(weatherlist.isEmpty())

return;

QList<WeatherData> list = weatherlist;

//qDebug()<<"count :"<<list.count();

//qDebug()<<"date :"<<list.at(0).date;

//qDebug()<<"type :"<<list.at(0).type;

while(1)

{

for(int i=0;i<list.count();i++)

{

ui->date\_show\_lb->setText(list.at(i).date);

ui->type\_text\_lb->setText(list.at(i).type);

QString typeIcon = QString("border-image:url(:/myimage/wImage/%1.png);").arg(list.at(i).type);

ui->type\_icon\_lb->setStyleSheet(typeIcon);

mSleep(1000);

}

}

}

如下图所示：



在menu.cpp文件的mSleep函数中，添加：

void menu::sleep(unsigned int msec)

{

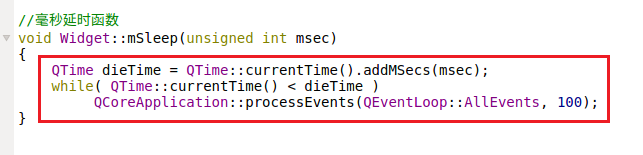
QTime dieTime = QTime::currentTime().addMSecs(msec);

while( QTime::currentTime() < dieTime )

QCoreApplication::processEvents(QEventLoop::AllEvents, 100);

}

如下图所示：



点击编译调试，效果演示：



可以看到正常获取到了天气数据并显示。

# 5. 功能拓展操作

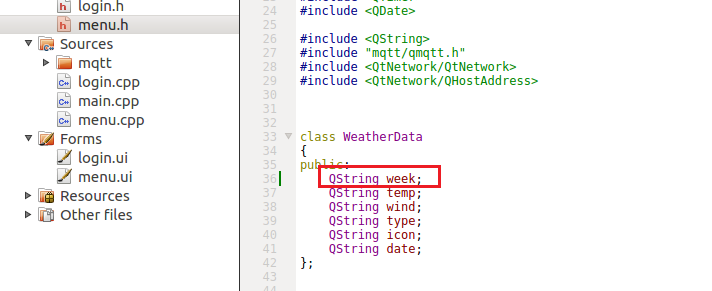
5.1 天气显示还存在问题的，请直接从返回的JSON数据中解析出日期与星期，并如下显示出来。



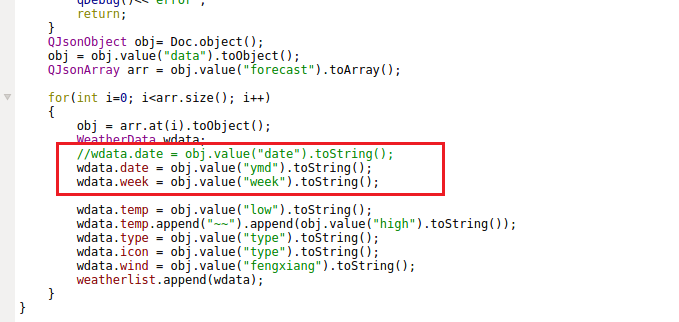
5.2 请将城市天气从广州改成佛山天气并显示出来。

5.1 操作如下：

1）打开menu.h文件



2）打开menu.cpp，在read\_weather函数，修改如下：



3）打开menu.cpp，在set\_weather\_info函数，修改如下：



重新编译执行，即可看到效果。