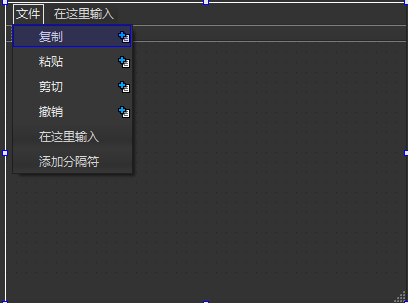
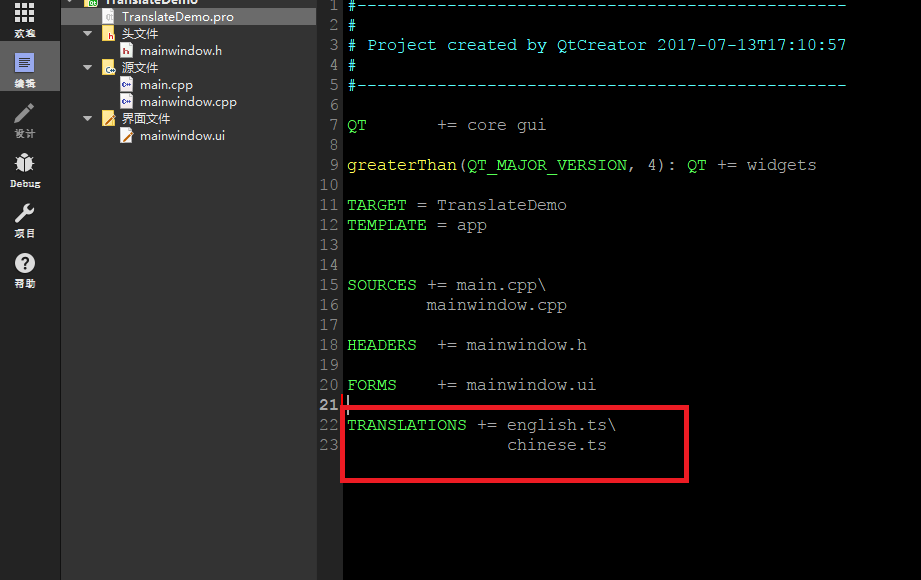
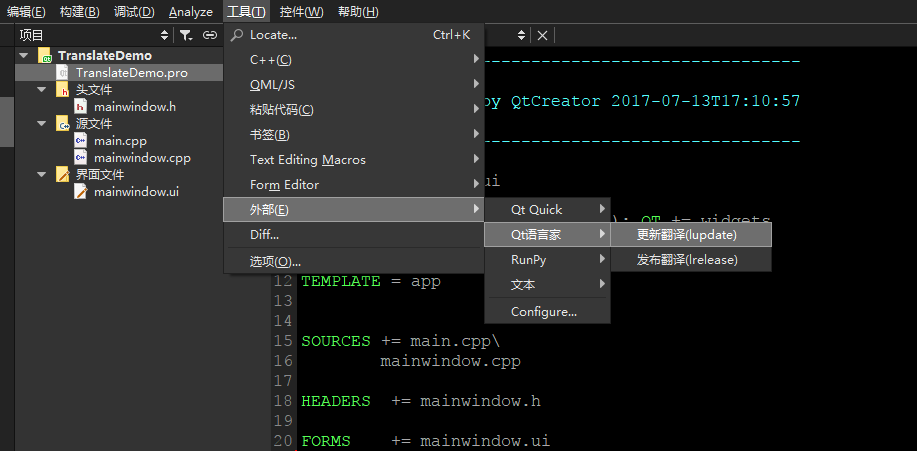
创建QT国际化例子



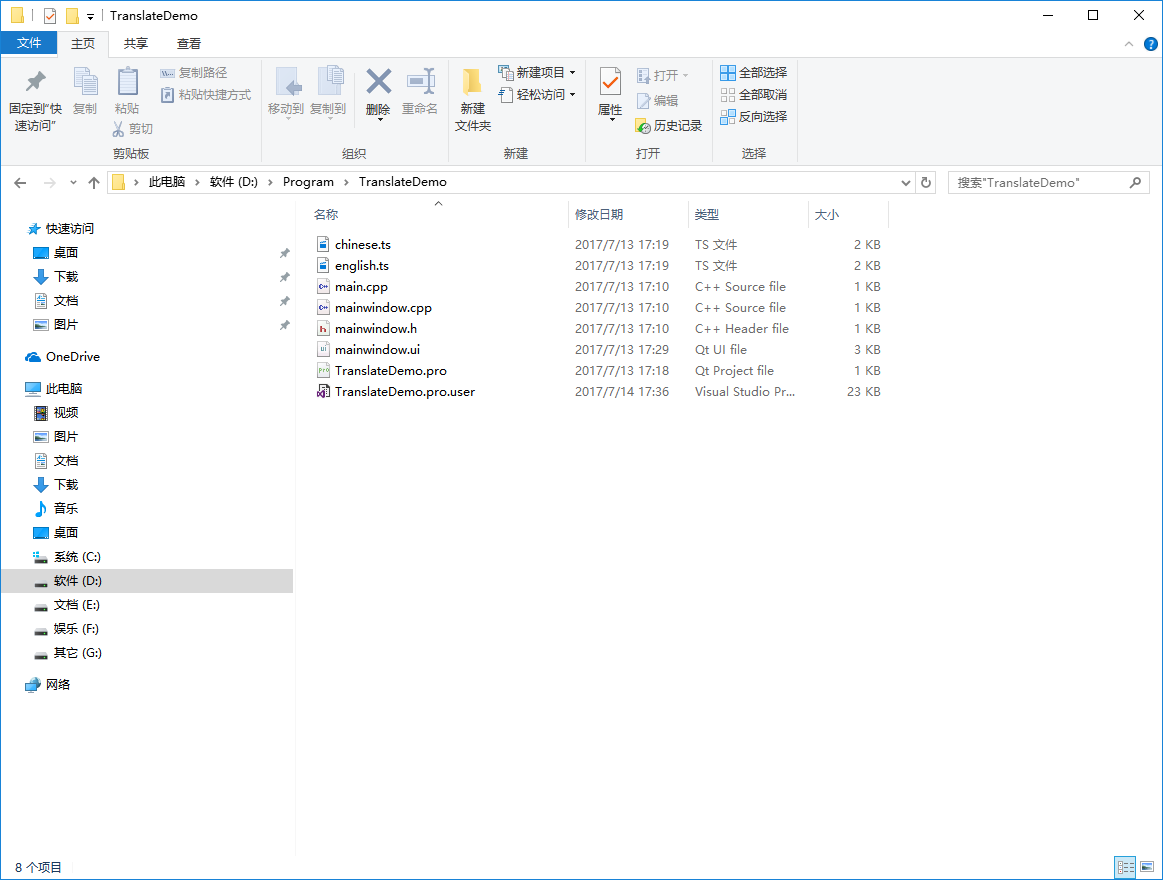
在TranslateDemo.pro中加入



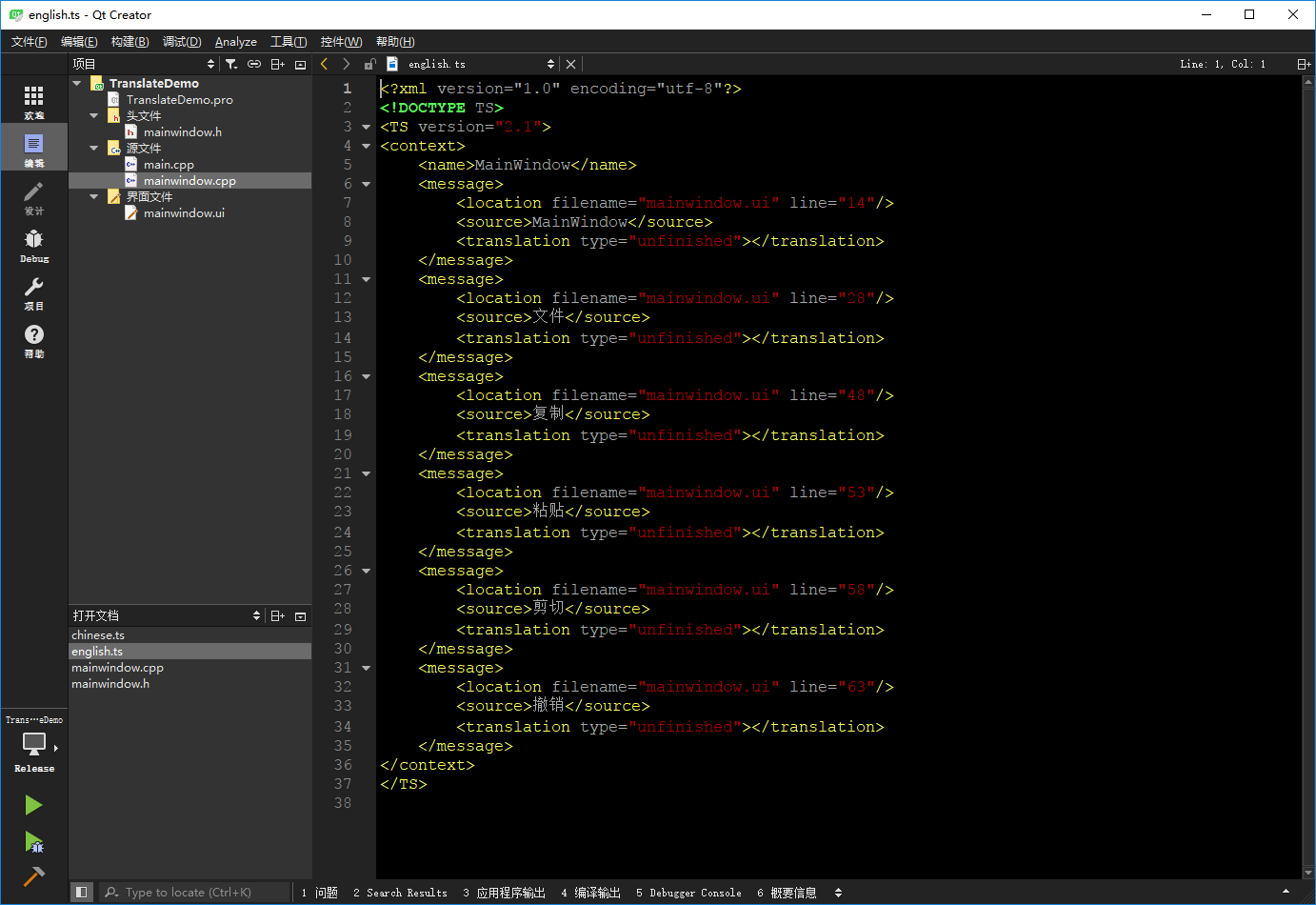
加入后保存.pro文件，然后使用Qt语言家更新翻译



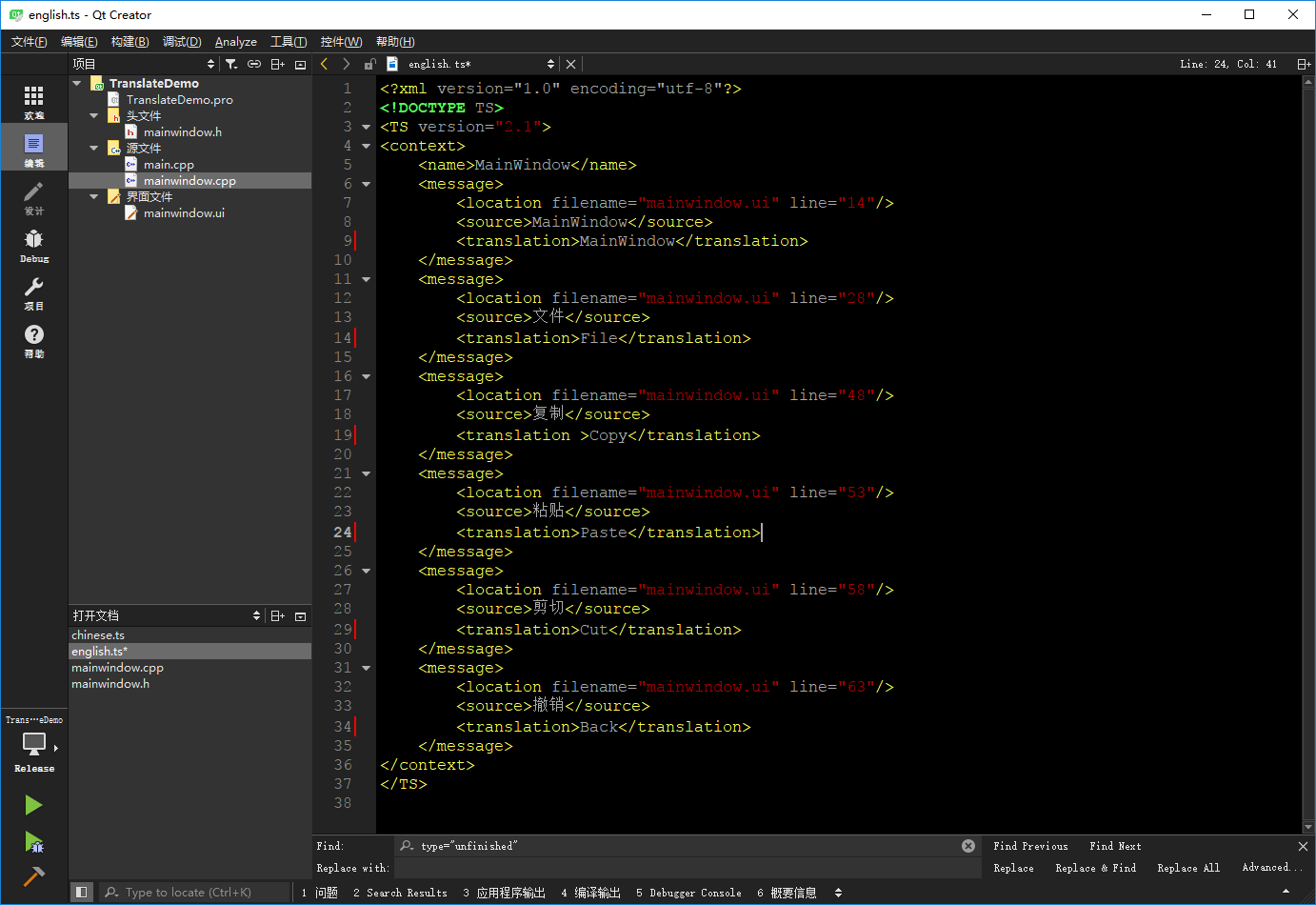
会生成两个ts文件，如下图



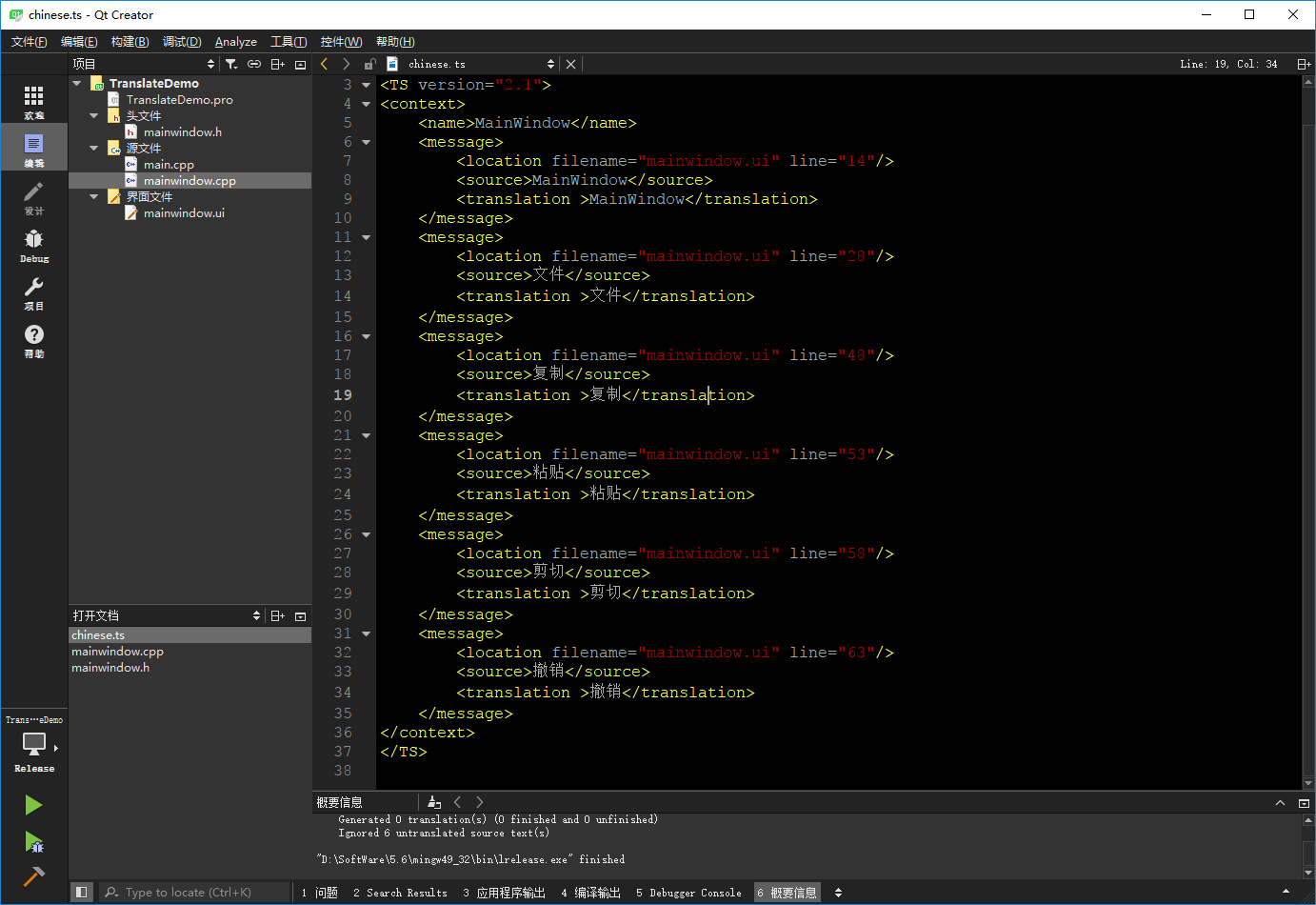
这两个ts文件可以直接用Qt以文本形式打开，或者使用Qt语言家Linguist打开，我更习惯用文本形式打开，打开english.ts是如下的一个xml格式的文件



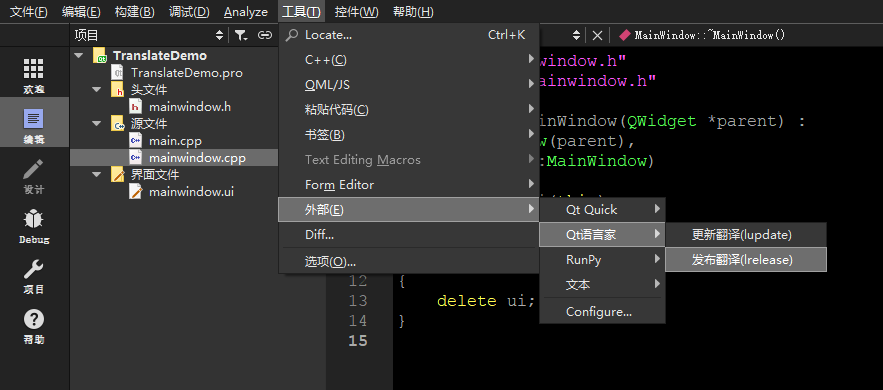
将其属性type = "unfinished"所有去掉，改为这种，点击保存文件



那个chinese也采用类似做法：如下图

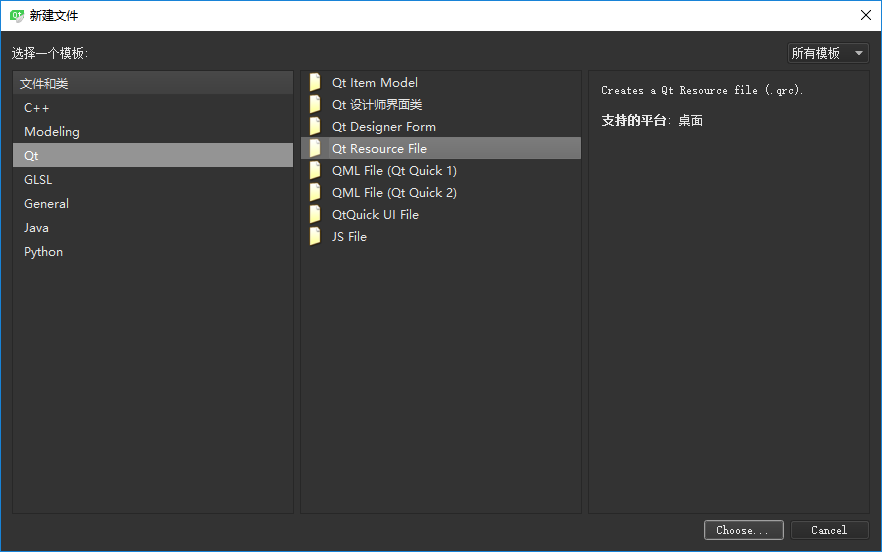


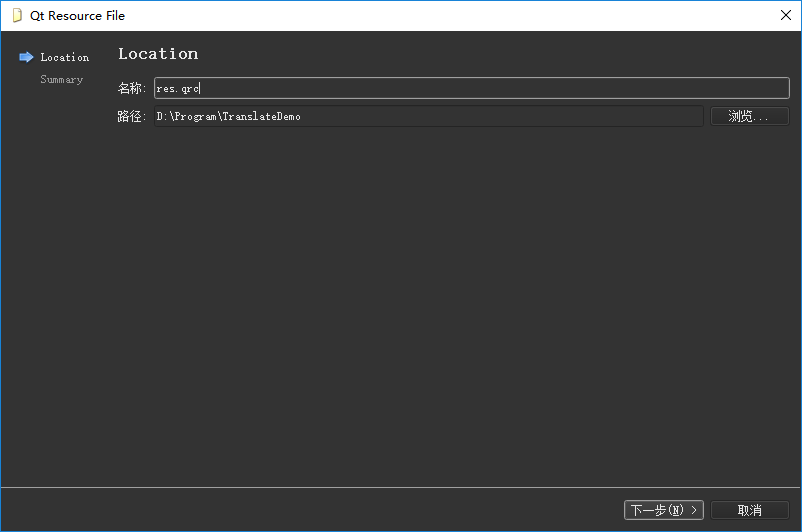
两个文件修改好后，点击保存，然后发布翻译



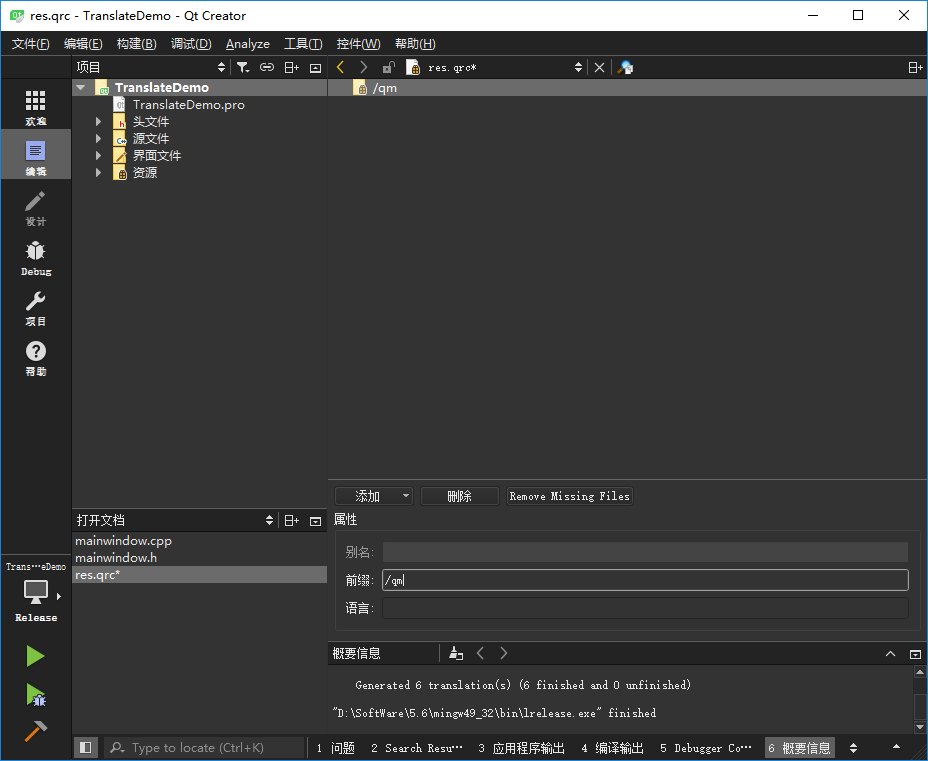
会生成两个qm文件，chinese.qm和English.qm文件

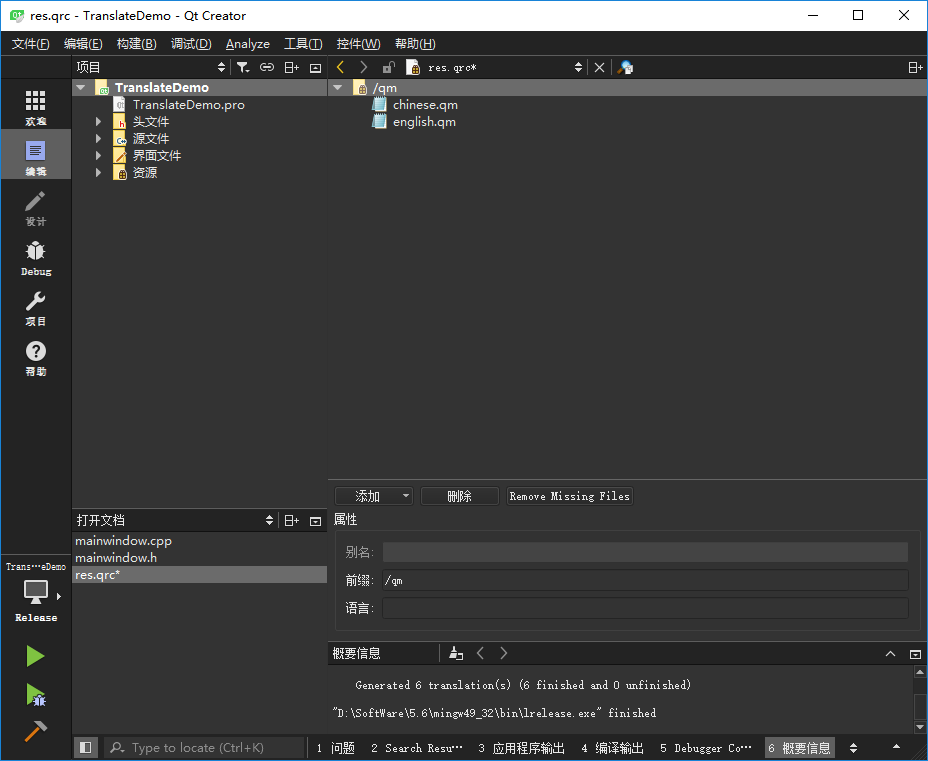
添加新文件res.qrc



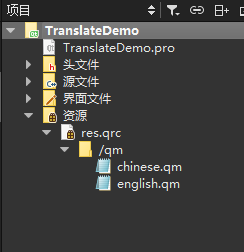


添加前缀qm





添加文件，将两个qm文件加进qrc中，如下图，然后保存



下面是代码：MainWindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QTranslator>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

private:

Ui::MainWindow \*ui;

QTranslator translater; //国际化

};

#endif // MAINWINDOW\_H

MainWindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~*MainWindow*()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

if (translater.load(":/qm/chinese.qm"))

{

qApp->installTranslator(&translater);

}

ui->retranslateUi(this); //这句必须带，在线实时切换

}

void MainWindow::on\_pushButton\_2\_clicked()

{

if (translater.load(":/qm/english.qm"))

{

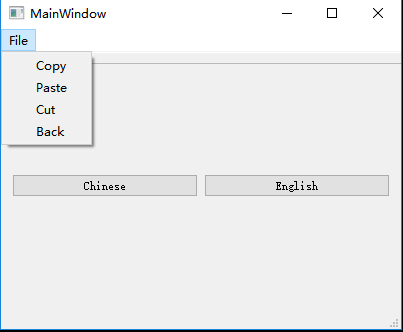
qApp->installTranslator(&translater);

}

ui->retranslateUi(this); //这句必须带，在线实时切换

}

效果：



再点击又回来了

