Designer设计器设计

Qt-程序入口

#include "desigerfileqt.h" //自定义界面头文件（创建一个创建同时创建.h, .cpp,.ui）

#include <QApplication> //Qt程序管理类

int main(int argc, char \*argv[])

{

QApplication a(argc, argv); //管理app类对象---一个qt程序只能创建一个QApplication对象

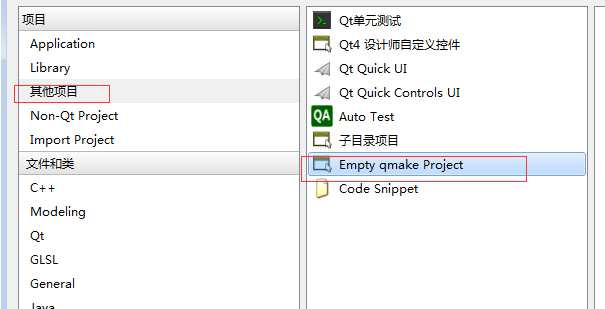
DesigerFileQt w; // 自定义窗口，

w.show();//显示窗口

return a.exec(); //qt程序进入循环（界面刷新， 动作处理）

}

创建一个空工程



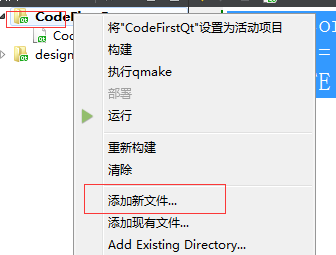
创建后修改工程文件

QT += core gui widgets

TARGET = CodeFirstQt

TEMPLATE = app

添加main.cpp QT程序入口文件



编写Qt程序框架

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

return a.exec();

}

编译运行

实现

#include <QApplication>

#include <QMainWindow>

#include <QPushButton>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

//创建窗口

QMainWindow w;

w.show();

//按钮

QPushButton bt1(&w);

bt1.setGeometry(10,20, 200,40);//定位按钮 --设置

bt1.setText("开灯");

bt1.show();

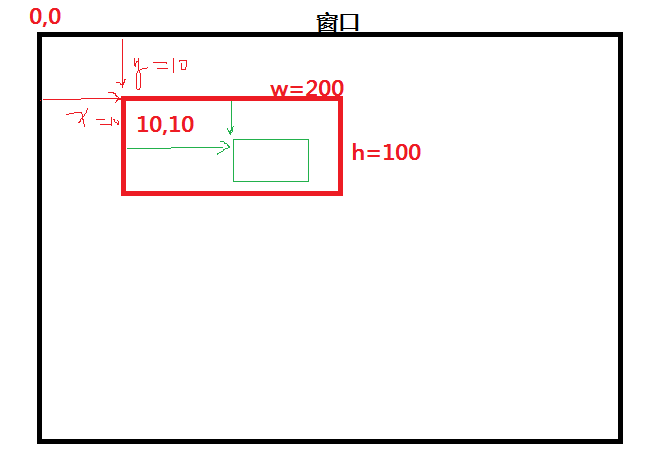
return a.exec();

}

----基本组件（控件）

按钮--QPushButton --

定位（坐标）



QRect rect = w.geometry();//获取位置尺寸

rect.setWidth(570);

rect.setHeight(310);

w.setGeometry(rect);

//w.resize(570,310);//从新设定大小

//按钮

QPushButton bt1(&w);

bt1.setGeometry(10,20, 200,100);//定位按钮

//设置或者文字显示

bt1.setText("开灯");

QString txt = bt1.text();

//设置文字字体大小

QFont font = bt1.font();

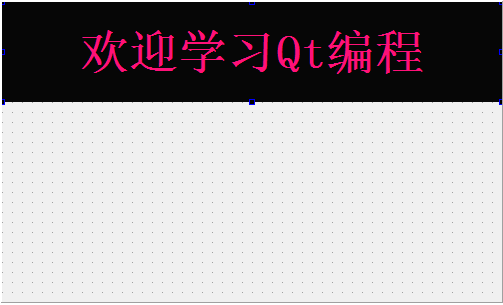
font.setPointSize(36);

bt1.setFont(font);

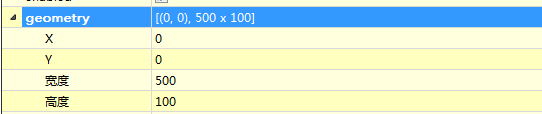
//设置颜色

bt1.setStyleSheet("color: rgb(255, 55, 202);");

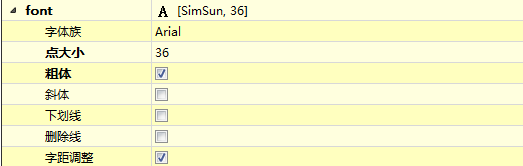
标签--QLabel



位置尺寸



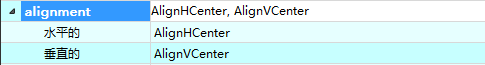
字体



字体，背景颜色



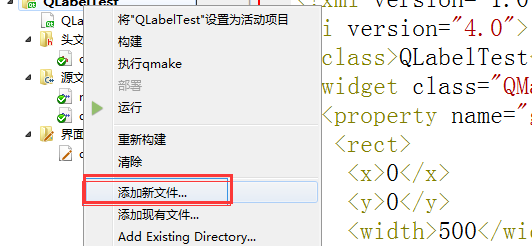
设置文字显示位置

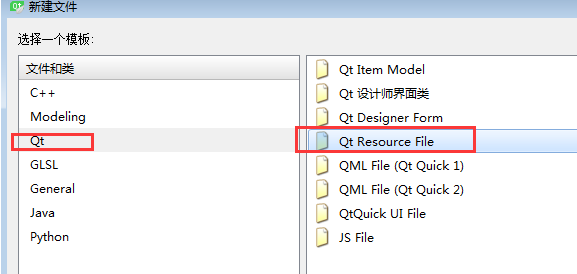


标签设置样式（背景图片， qss）

准备好图片放在工程目录下（可以是工程目录的目录）

在集成开发环境中创建一个资源文件image.qrc





添加前缀， 添加文件， 保存

设置图片



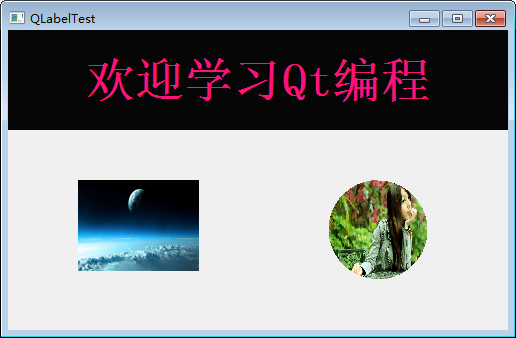
设置样式



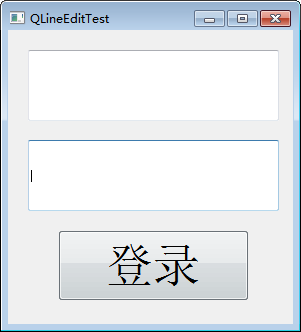
border-image: url(:/new/prefix1/image/girl.jpg);

border:1px solid#123456;

border-radius:50px;



单行输入：QLineEdit



提示输入



设置回显模式为密码显示



设置输入的字符长度



设置文字显示位置



样式设置--设置全局（在最外面窗口上设置）

QMainWindow#QLineEditTest{background-image: url(:/new/prefix1/image/girl.jpg);}

QLineEdit{

border:1px solid#123455;

border-radius:15px;

}

QPushButton{

border:1px solid#123455;

border-radius:15px;

}

QPushButton:hover{

background-color: rgb(52, 16, 255);

}

QPushButton:pressed {

background-color: rgb(255, 41, 216);

}

如果界面继承是QWidget那么设置主窗口背景图片（放构造函数）

QPalette palette;

QPixmap pixmap(":/new/prefix1/image/girl.jpg");

QBrush brush(pixmap);

palette.setBrush(QPalette::Background, brush);

this->setPalette(palette);

