## 1.Qtime，使用qsrand函数需要用到

|  |
| --- |
| #include<QTime>  #include<QDebug>  void Widget::**testQtimeWithqsrand**()  {  qsrand((unsigned int)QTime(0,0,0).secsTo(QTime::currentTime()));  for (int i=0;i<10;i++) {  qDebug()<<qrand()%11-1<<endl;  }  } |

## 2.Qtimer，qt的定时器：

创建一个demo工程，

在Widget.h中添加一个Qtimer类的成员变量m\_ptimer

|  |
| --- |
| #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  #include<QTimer>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  QTimer \*m\_ptimer;  private slots:  void **on\_btnStart\_clicked**();//Start按钮的槽函数  void **on\_btnStop\_clicked**();//Stop按钮的槽函数  void **onTimer**();//Qtimer的槽函数  private:  Ui::Widget \*ui;  void **testQtimeWithqsrand**();  };  #endif // WIDGET\_H |

界面如下

|  |
| --- |
|  |

为两个按钮添加槽函数：

|  |
| --- |
|  |

和Qtimer的槽函数

|  |
| --- |
| void Widget::**onTimer**()  {  int num =qrand()%100;  ui->lcdNumber->display(QString::number(num));  } |

实现的功能是当用户点击Start按钮，LCDNumber控件显示随机生成的数字，

当用户点击Stop按钮，就停止显示

Widget.cpp的源码：

|  |
| --- |
| #include "widget.h"  #include "ui\_widget.h"  #include<QTime>  #include<QDebug>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  //创建定时器  m\_ptimer = new QTimer(this);  connect(m\_ptimer,&QTimer::timeout,this,&Widget::onTimer);  //testQtimeWithqsrand();  // ui->lcdNumber->display("12");  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::**testQtimeWithqsrand**()  {  qsrand((unsigned int)QTime(0,0,0).secsTo(QTime::currentTime()));  for (int i=0;i<10;i++) {  //qDebug()<<qrand()%10<<endl;//产生0-9随机数  //qDebug()<<qrand()%100<<endl;//产生0-99随机数  //qDebug()<<qrand()%1000<<endl;//产生0-999随机数  int num =qrand()%100;  ui->lcdNumber->display(QString::number(num));  }  }  void Widget::**on\_btnStart\_clicked**()  {  m\_ptimer->start(1000);//这个函数需要传递一个时间间隔否则显示的非常快，看不清数字  }  void Widget::**on\_btnStop\_clicked**()  {  m\_ptimer->stop();  }  void Widget::**onTimer**()  {  int num =qrand()%100;  ui->lcdNumber->display(QString::number(num));  } |

## 3.Qimage和Qpixmap

## 4.QLabel：注意Qlabel有一个setPixmap方法可以显示图片

|  |
| --- |
| //QImage img("fun.jpg");  //ui->label\_pic->setPixmap(QPixmap::fromImage(img));  //写法2  QPixmap pix("fun.jpg");//不支持png格式，返回null  QPixmap pix2 = pix.scaled(ui->label\_pic->size());  ui->label\_pic->setPixmap(pix2); |

## 5.Qmovie类，只可以播放gif图片,在头文件中声明Qmovie m\_pmovie;

|  |
| --- |
| m\_pmovie = new QMovie("./dogcr.gif");//QMovie只能播放gif如果需要播放视频，使用QMediaPlayer类和QVideoWidgetQMediaPlayerList  ui->label->setMovie(m\_pmovie);  m\_pmovie->start();//开始播放  m\_pmovie->stop();//停止播放 |

## 6.QMediaPlayer类和QVideoWidget和QmediaPlayerList播放视频

## 7.QMessageBox

创建一个简单工程，如下：

|  |  |
| --- | --- |
|  |  |
| 主要代码，那两个按钮是槽函数  #include "widget.h"  #include "ui\_widget.h"  #include<QMessageBox>  #include<QDebug>  #include<qdesktopservices.h>  #include<QString>  #include<QUrl>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::**on\_btnHello\_clicked**()  {  QString btnName;  btnName = ui->btnHello->*metaObject*()->className();  int k =QMessageBox::information(this,"hello","hello "+btnName,QMessageBox::Ok|QMessageBox::Cancel|QMessageBox::Help,QMessageBox::Ok);  if(k==QMessageBox::Ok)  {  qDebug()<<"QMessageBox::Ok clicked"<<endl;  }  else if(k==QMessageBox::Cancel)  {  qDebug()<<"QMessageBox::Cancel clicked"<<endl;  }  else if(k==QMessageBox::Help){  QString url="www.baidu.com";  QDesktopServices::openUrl(QUrl(url.toLatin1()));//Qt打开网页  }  }  void Widget::**on\_btnContent\_clicked**()  {  QString content = ui->textEdit->toPlainText();  if(content.trimmed()=="")  {  QMessageBox::warning(this,"warning","no content to show ");  ui->textEdit->setFocus();  }  else {  QMessageBox::information(nullptr,"content",content);  }  } |  |

**注意，在qt中表示内容为空是””,“ ”不表示内容为空因为空格也是字符**

另外hello按钮还有一个打开网页的功能，当用户点击help按钮，打开百度。注意需要包含的是#include<qdesktopservices.h>

这个头文件有QDesktopServices类

## 8，QdesktopServices类

## 9.Qurl类

## 10 QColor，QColorDialog，Qpalette，QFont，QfontDialog

实例：

|  |  |
| --- | --- |
|  |  |
| //widget.h  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  private slots:  void **on\_btnColor\_clicked**();  void **on\_btnFont\_clicked**();  void **on\_btnOpen\_clicked**();  void **on\_btnSave\_clicked**();  private:  Ui::Widget \*ui;  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  #include<QColorDialog>  #include<QFontDialog>  #include<QFileDialog>  #include<QColor>  #include<QFont>  #include<QFile>  #include<QIODevice>  #include<QMessageBox>  #include<QDebug>  #include<QPalette>  #include<QTextStream>  #include<QByteArray>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::**on\_btnColor\_clicked**()  {  //修改richedit编辑框的文本颜色  //直接设置字体颜色，只对点击按钮之后的文本颜色，之前的需要选中然后点击按钮  QString content = ui->textEdit->toPlainText().trimmed();  QColorDialog dlg(this);  QColor color =dlg.getColor();  //设置编辑框的文本颜色有2种方法：  //1.方法1，使用调色板，这种方法比较适合标签对象  QPalette txt\_platte = ui->textEdit->palette();  txt\_platte.setColor(QPalette::Text,color);//注意这里是Text  ui->textEdit->setPalette(txt\_platte);  //方法2，使用编辑框的setTextColor方法，该方法对设置去的文本没有作用，必须选中重新设置，比较实用  // QString selected = ui->textEdit->textCursor().selectedText().trimmed();//qt获取选中文本  // ui->textEdit->setTextColor(color);  // if(content==""){  // QMessageBox::information(nullptr,"说明","设置颜色成功,输入文本试试");  // ui->textEdit->setFocus();  // }  // else  // {  // QMessageBox::information(nullptr,"注意","old text need to reset color");  // ui->textEdit->setFocus();  // }  //标签对象设置文本颜色只有1种方法  QPalette lbl\_palette =ui->lbl\_test->palette();//获取标签的调色板  lbl\_palette.setColor(QPalette::WindowText,color);//利用调色板设置颜色，WindowText  ui->lbl\_test->setPalette(lbl\_palette);//标签对象重新设置调色板  }  void Widget::**on\_btnFont\_clicked**()  {  QFontDialog dlg(this);  bool ok;  QFont font = dlg.getFont(&ok);  ui->textEdit->setFont(font);  }  void Widget::**on\_btnOpen\_clicked**()  {  QString filename = QFileDialog::getOpenFileName();  QFile file(filename);  if(!file.*open*(QIODevice::ReadOnly|QIODevice::Text))  {  return;  }  QTextStream in(&file);  QString content="";  while (!in.atEnd()) {  content += in.readLine();  }  file.*close*();  ui->textEdit->setText(content);  }  void Widget::**on\_btnSave\_clicked**()  {  QString content = ui->textEdit->toPlainText();  if(content=="")  {  QMessageBox::critical(nullptr,"error","Please Enter Content");  ui->textEdit->setFocus();  }  else {  QFile file(QFileDialog::getSaveFileName());  if(!file.*open*(QIODevice::WriteOnly|QIODevice::Text))  {  return;  }  //QTextStream out(&file);  QByteArray arr = content.toUtf8();  file.write(arr,arr.length());  file.*close*();  QMessageBox::information(nullptr,"结果","保存成功");  }  } |

## 11.QInputDialog，有很多获取输入的方法，都是静态方法实例如下

|  |  |
| --- | --- |
|  |  |
| //头文件  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  private slots:  void **on\_btn\_getAge\_clicked**();  void **on\_btn\_getWeight\_clicked**();  void **on\_btn\_getItem\_clicked**();  void **on\_btn\_getText\_clicked**();  private:  Ui::Widget \*ui;  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  #include<QInputDialog>  #include<QString>  #include<QStringList>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::**on\_btn\_getAge\_clicked**()  {  bool ok;  int age =QInputDialog::getInt(this,"input your age","Age",10,1,100,1,&ok);  if(ok)  {  ui->ed\_age->setText(QString::number(age));  }  }  void Widget::**on\_btn\_getWeight\_clicked**()  {  bool ok;  double weight =QInputDialog::getDouble(this,"input your weight","Weight",50.0,1.0,1000.0,1.0,&ok);  if(ok)  {  ui->ed\_weight->setText(QString::number(weight));  }  }  void Widget::**on\_btn\_getItem\_clicked**()  {  QStringList items;  bool ok;  items<<tr("语文，数学，英语")<<tr("物理,化学,英语")<<tr("语文,电脑,英语")<<tr("绘画,数学,英语,电工");  QString item = QInputDialog::getItem(this,"please choose your subject","Subjects",items,0,false,&ok);  if(ok)  {  ui->ed\_subject->setText(item);  }  }  void Widget::**on\_btn\_getText\_clicked**()  {  QString text;  bool ok;  text = QInputDialog::getText(this,"introduce","please say something about yourself",QLineEdit::Normal,"",&ok);  if(ok && text.trimmed()!="")  {  ui->ed\_info->setText(text);  }  } |

12.QProcessDialog，实例

|  |  |
| --- | --- |
|  |  |
| //widget.h  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  #include<QProgressDialog>  #include<QTimer>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  private slots:  void **on\_pushButton\_clicked**();  void **onCancel**();  void **onTimer**();  private:  Ui::Widget \*ui;  QProgressDialog\* m\_pdlg;//成员变量  QTimer\* m\_ptimer; //成员变量  int m\_prog; //成员变量  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  #include<QDebug>  #include<Windows.h>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  m\_pdlg = nullptr;  m\_prog=0;  m\_ptimer = new QTimer(this);  connect(m\_ptimer,&QTimer::timeout,this,&Widget::onTimer);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::**on\_pushButton\_clicked**()  {  if(m\_pdlg ==nullptr)  {  m\_pdlg = new QProgressDialog(this);  m\_pdlg->setRange(0,100);  m\_pdlg->setAutoClose(false);  m\_pdlg->setWindowModality(Qt::WindowModal);  m\_pdlg->show();  }  else{  m\_pdlg->show();  }  connect(m\_pdlg,&QProgressDialog::canceled,this,&Widget::onCancel);  m\_ptimer->start();  }  void Widget::**onCancel**()  {  m\_ptimer->stop();  qDebug()<<"Operation canceled..."<<endl;  m\_prog = 0;//记住这里需要清零，否则再次点击就没有效果  }  void Widget::**onTimer**()  {  m\_prog++;  m\_pdlg->setValue(m\_prog);  Sleep(100);  if(m\_prog==100)  m\_ptimer->stop();    } |

效果

|  |
| --- |
|  |

## 12 QErrorMessage类

|  |
| --- |
|  |

## 13、QMouseEvent，

一个比较有趣的实例：

|  |  |
| --- | --- |
|  |  |
| //widget.h  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  protected:  void ***mousePressEvent***(QMouseEvent \*event);  void ***mouseMoveEvent***(QMouseEvent \*event);  private:  Ui::Widget \*ui;  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  #include<QDebug>  #include<QMouseEvent>  #include<QPalette>  #include<QColorDialog>  #include<QColor>  #include<QKeyEvent>  #include<QFontDialog>  #include<QFont>  #include<QString>  #include<QStringList>  #include<QInputDialog>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::***mousePressEvent***(QMouseEvent \*event)  {  //qDebug()<< event->globalPos()<<endl;  //qDebug()<< event->windowPos()<<endl;  if(event->button()==Qt::LeftButton) //注意Qt中鼠标左键的写法  {  //  int x = event->globalX();  int y = event->globalY();  ui->ed\_x->setText(QString::number(x));  ui->ed\_y->setText(QString::number(y));  int wx = int(event->windowPos().x());//注意类型转换的新写法，构造函数法  int wy = int(event->windowPos().y());  ui->ed\_x\_win->setText(QString::number(wx));  ui->ed\_y\_win->setText(QString::number(wy));  }  if(event->button()==Qt::RightButton) //注意Qt中鼠标右键的写法  {  QStringList items;  items << tr("改变窗体颜色")<<tr("改变文本字体");  bool ok;  QString item = QInputDialog::getItem(this,"请选择","操作选项",items,0,false,&ok);//获取用户选择，注意QInputDialog的用法  if(ok){  if(item == "改变窗体颜色")  {  QColor color = QColorDialog::getColor();  QPalette palette = this->palette();  palette.setColor(QPalette::Window,color);  this->setPalette(palette);  }  else if (item == "改变文本字体") {  QFont font =QFontDialog::getFont(&ok);  if(ok){  ui->lbl\_sam->setFont(font);  }  }  }  }  }  void Widget::***mouseMoveEvent***(QMouseEvent \*event)  {  int x = event->globalX();  int y = event->globalY();  ui->ed\_x->setText(QString::number(x));  ui->ed\_y->setText(QString::number(y));  int wx = int(event->windowPos().x());  int wy = int(event->windowPos().y());  ui->ed\_x\_win->setText(QString::number(wx));  ui->ed\_y\_win->setText(QString::number(wy));  } |

|  |  |
| --- | --- |
|  |  |

## 14，QpaintEvent ，Qpainter，QpaintDevice，QPen，QRect，QPoint，

一个实例，不是非常好，可以继续改进

|  |  |
| --- | --- |
|  |  |
| //widget.h  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  #include<QMouseEvent>  #include<QPaintEvent>  #include<QPoint>  #include<QPixmap>  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  QPoint m\_startPt,m\_endPt;  QPixmap m\_pix,m\_tmpPix;  bool m\_isDrawing;  protected:  void ***mousePressEvent***(QMouseEvent \*event);  void ***mouseMoveEvent***(QMouseEvent \*event);  void ***mouseReleaseEvent***(QMouseEvent \*event);  void ***paintEvent***(QPaintEvent \*event);  private:  Ui::Widget \*ui;  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  #include<QPainter>  #include<QPen>  #include<QColor>  #include<QDebug>  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  resize(600,500);  m\_isDrawing = false;  m\_pix = QPixmap(300,300);  m\_pix.fill(Qt::white);  // ui->lbl\_paint->setPixmap(m\_pix);  }  Widget::~***Widget***()  {  delete ui;  }  void Widget::***mousePressEvent***(QMouseEvent \*event)  {  if(event->button()==Qt::LeftButton)  {  m\_startPt = event->pos();  m\_isDrawing = true;  }  }  void Widget::***mouseMoveEvent***(QMouseEvent \*event)  {  if(event->button()==Qt::LeftButton)  {  m\_endPt = event->pos();  update();//引发窗口重绘  }  }  void Widget::***mouseReleaseEvent***(QMouseEvent \*event)  {  if(event->button()==Qt::LeftButton)  {  m\_endPt = event->pos();  m\_isDrawing = false;  update();//引发窗口重绘  }  }  void Widget::***paintEvent***(QPaintEvent \*event)  {  int x,y,w,h;  x = m\_startPt.x();  y = m\_startPt.y();  w = m\_endPt.x()-x;  h = m\_endPt.y()-y;  //创建绘画者  QPainter painter(this);  if(m\_isDrawing)  {  //保存以前的内容  m\_tmpPix = m\_pix;  QPen pen(QColor(255,0,255));  QPainter pter(&m\_tmpPix);//创建第二个绘画者  pter.setPen(pen);  pter.drawRect(x,y,w,h);  int left = ui->lbl\_paint->x();  int top = ui->lbl\_paint->y();  qDebug()<<left<<","<<top<<endl;  //ui->lbl\_paint->setPixmap(m\_tmpPix);  painter.setPen(pen);  painter.drawPixmap(left,top,m\_tmpPix);  }  else {  QPainter pter(&m\_pix);//创建第二个绘画者  QPen pen(QColor(255,0,255));  pter.setPen(pen);  pter.drawRect(x,y,w,h);  int left = ui->lbl\_paint->x();  int top = ui->lbl\_paint->y();  //ui->lbl\_paint->setPixmap(m\_pix);  painter.setPen(pen);  painter.drawPixmap(left,top,m\_pix);  }  } |

## 15.QPushButton

实例：定义一个类从QPushButton派生，实现移动鼠标就改变按钮的位置

|  |  |
| --- | --- |
|  |  |
| //custombutton.h  #ifndef CUSTOMBUTTON\_H  #define CUSTOMBUTTON\_H  #include <QWidget>  #include<QPushButton>  #include<QMouseEvent>  class **CustomButton** : public QPushButton  {  Q\_OBJECT  public:  explicit **CustomButton**(QPushButton \*parent = nullptr);  signals:  public slots:  protected:  void ***mouseMoveEvent***(QMouseEvent\* event);  void ***mousePressEvent***(QMouseEvent\* event);  };  #endif // CUSTOMBUTTON\_H | //custombutton.cpp  #include "custombutton.h"  #include<QTime>  #include<QDebug>  #include<QMessageBox>  CustomButton::**CustomButton**(QPushButton \*parent) : QPushButton(parent)  {  setMouseTracking(true);//设置自动捕获鼠标，这样子在鼠标移动事件中就不需要先按下鼠标左键，直接移动就可以了  qsrand(uint(QTime(0,0,0).secsTo(QTime::currentTime())));//产生随机数种子  }  void CustomButton::***mouseMoveEvent***(QMouseEvent \*event)  {  //写法一，比较啰嗦  // int btn\_x = this->x();  // int btn\_y = this->y();  // if((event->x()+btn\_x)>= this->x() && ((event->x()+btn\_x) <= this->x()+this->width()))  // {  // if((event->y()+btn\_y)>= this->y() && ((event->y()+btn\_y)<=this->y()+this->height()))  // {  // QWidget\* parent = this->parentWidget();  // int x = qrand()%(parent->width() - this->width());  // int y = qrand()%(parent->height() - this->height());  // this->move(x,y);  // //this->update();  // }  // }  //写法2，比较简单因为现在是按钮类的鼠标移动事件，不需要太多的判断  QWidget\* parent = this->parentWidget();  int x = qrand()%(parent->width() - this->width());  int y = qrand()%(parent->height() - this->height());  //qDebug()<< x <<","<<y<<endl;  // QMessageBox::information(nullptr,"point",QString::number(x)+","+QString::number(y));  this->move(x,y);  }  void CustomButton::***mousePressEvent***(QMouseEvent \*event)  {  qDebug()<< event<<endl;  //如果鼠标移动事件有效，下面的代码无法执行，因为你根本无法点击到按钮  // if(event->button() == Qt::LeftButton)  // {  // QWidget\* parent = this->parentWidget();  // int x = qrand()%(parent->width() - this->width());  // int y = qrand()%(parent->height() - this->height());  // //qDebug()<< x <<","<<y<<endl;  // // QMessageBox::information(nullptr,"point",QString::number(x)+","+QString::number(y));  // this->move(x,y);  // }  } |
| //widget.h  #ifndef WIDGET\_H  #define WIDGET\_H  #include <QWidget>  #include"custombutton.h"  QT\_BEGIN\_NAMESPACE  namespace **Ui** { class **Widget**; }  QT\_END\_NAMESPACE  class **Widget** : public QWidget  {  Q\_OBJECT  public:  **Widget**(QWidget \*parent = nullptr);  ~***Widget***();  private:  Ui::Widget \*ui;  CustomButton\* m\_pbtn;  };  #endif // WIDGET\_H | //widget.cpp  #include "widget.h"  #include "ui\_widget.h"  Widget::**Widget**(QWidget \*parent)  : QWidget(parent)  , ui(new Ui::Widget)  {  ui->setupUi(this);  m\_pbtn = new CustomButton();  m\_pbtn->setText("Testing");  m\_pbtn->setParent(this);  // m\_pbtn->setGeometry(this->x(),this->y(),100,60);  m\_pbtn->show();  }  Widget::~***Widget***()  {  delete ui;  delete m\_pbtn;  } |