# QT\_NOTE

//去掉标题栏

this->setWindowFlags(Qt::FramelessWindowHint);

//设置透明2-窗体标题栏不透明,背景透明

this->setAttribute(Qt::WA\_TranslucentBackground, true);

//隐藏鼠标

this->setCursor(Qt::BlankCursor);

//qt屏幕翻转

~ # ./qtText -qws -display "Transformed:Rot270" &

**1）显示不规则图标：**

#include <QPixmap>

#include <QBitmap>

QPixmap maskRound;

maskRound.load("28.png");

m\_labelRound = new QLabel(this);

m\_labelRound->move(10, 20);

m\_labelRound->setFixedSize(QSize(60, 60));

m\_labelRound->setMask(maskRound.mask());

m\_labelRound->setPixmap(QPixmap("28.png"));



**2）arm添加中文字库：**

添加字库到：qt4.7.1/lib/fonts/ wenquanyi \*.dpf

QTextCodec::setCodecForTr(QTextCodec::codecForName("utf8"));  
    QTextCodec::setCodecForCStrings(QTextCodec::codecForName("utf8"));

 QFont font("wenquanyi",12,QFont::Bold); //使用wenquanyi字体  
      this->setFont(font);

**3)定时器出发按键**

btn = new QPushButton(this);

btn->setText("示汉字");

connect(btn,SIGNAL(clicked()),this,SLOT(slot\_string()));

timer=new QTimer(this);

//connect(timer,SIGNAL(timeout()),this,SLOT(timeoutslot()));//tiemoutslot()为自定义槽

connect(timer,SIGNAL(timeout()), this,SLOT(timeoutslot()));

timer->start(1000);

Widget::timeoutslot(){

emit btn->click();

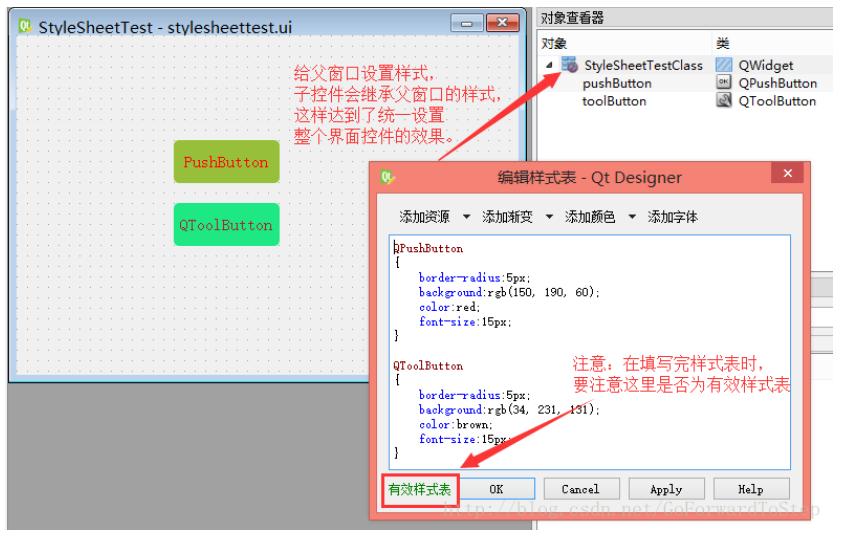
}

void Widget::slot\_string(){

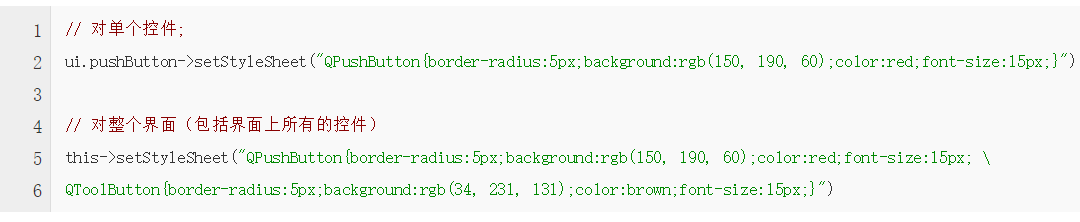
qDebug()<< "123";

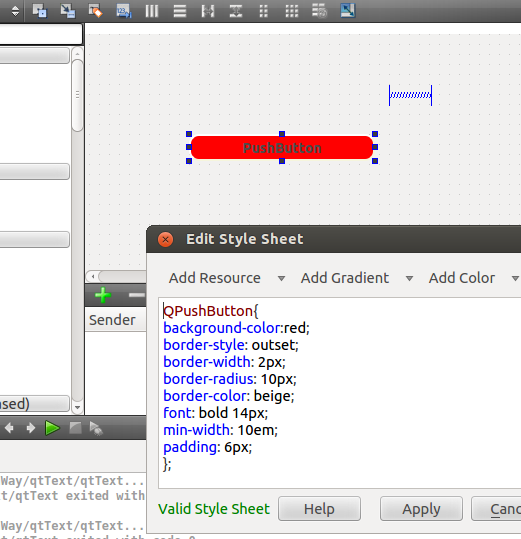
}

**4）样式表设置：**



在代码中添加：





//设置Qlable背景透明,圆角

QLabel \*label=new QLabel();

label->setStyleSheet("QLabel{background-color: transparent;border-radius:10px;padding:2px 4px;}");

**旋转90°：**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

se\_terminal 显示窗口

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

se\_terminal \*form = new se\_terminal;

QGraphicsScene \*scene = new QGraphicsScene;

scene->addWidget(form);

QGraphicsView \*view = new QGraphicsView(scene);

view->resize(720,1280);

view->rotate(270);

view->setStyleSheet("border:0px");

view->setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

view->setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

view->setWindowFlags(Qt::FramelessWindowHint);

view->show();

**QT-QLable旋转：**

QMatrix leftmatrix;

leftmatrix.rotate(90);

qlable = new QLabel(this);

qlable->resize(50,200);

qlable->move(20,30)；

qlable->setPixmap(QPixmap("1.png").transformed(leftmatrix,Qt::*SmoothTransformation*));

**5）取消信号和槽：**

**disconnect(btn,SIGNAL(clicked()),this,SLOT(slot\_string()));**

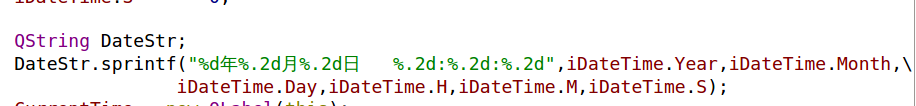
。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。。

元对象系统：

1. 基类QObject：任何需要使用元对象系统功能的类必须继承QObject。
2. Q\_OBJECT宏：Q\_OBJECT宏必须出现在类的私有声明区，用于启动元对象的特征
3. 元对象编辑器：为QObject子类实现元对象特性提供必要的代码实现。



**Qt Int，float，double To QString：**



**Qt: StyleSheet**

Qt中设置按钮或QWidget的外观是，可以使用QT Style Sheets来进行设置，非常方便。可以用

setStyleSheet("font: bold; font-size:20px; color: rgb(241, 70, 62); background-color: green");

来进行设置，其他的样式介绍如下：

font: bold; 是否粗体显示  
border-image:""; 用来设定边框的背景图片。  
border-radius:5px; 用来设定边框的弧度。可以设定圆角的按钮  
border-width: 1px； 边框大小

font-family:""; 来设定字体所属家族，  
font-size:20px; 来设定字体大小  
font-style:""; 来设定字体样式  
font-weight:20px; 来设定字体深浅

background-color: green; 设置背景颜色  
background:transparent; 设置背景为透明  
color:rgb(241, 70, 62); 设置前景颜色  
selection-color:rgb(241, 70, 62); 用来设定选中时候的颜色  
可以使用border-top，border-right，border-bottom，border-left分别设定按钮的上下左右边框，  
同样有border-left-color, border-left-style, border-left-width.等分别来设定他们的颜色，样式和宽度

**QT\_Painter事件**

**抗锯齿：**

QPainter painter(this);

painter.setPen(QPen(Qt::black, 5, Qt::DashDotLine, Qt::RoundCap));

painter.setBrush(Qt::yellow);

painter.drawEllipse(50, 150, 200, 150);

painter.setRenderHint(QPainter::Antialiasing, true);// 将Antialiasing属性（也就是反走样）设置为true

painter.setPen(QPen(Qt::black, 5, Qt::DashDotLine, Qt::RoundCap));

painter.setBrush(Qt::yellow);

painter.drawEllipse(300, 150, 200, 150);

**线性渐变：**

1.void QGradient::setColorAt ( qreal position, const QColor & color )

这个函数的作用是，把 position 位置的颜色设置成 color。其中，position 是一个 [0, 1] 闭区间的数字。也就是说，position 是相对于我们建立渐变对象时做的那个起始点和终止点区间的一个比例。以这个线性渐变为例，在从 (60, 50) 到 (200, 200) 的线段上，在 0.2，也就五分之一处设置成白色，在 0.6 也就是五分之三处设置成绿色，在 1.0 也就是终点处设置成黑色。创建 QBrush 对象时，把这个渐变对象传递进去，然后就可以运行了。

2.QConicalGradient::QConicalGradient ( qreal cx, qreal cy, qreal angle )

前两个参数 cx 和 cy 组成角度渐变的中心点，第三个参数是渐变的起始角度。在我们的例子中，我们将渐变中心点设置为 (0, 0)，起始角度为 0。类似线性渐变，角度渐变的 setColorAt() 函数同样接受两个参数，第一个是角度比例，第二个是颜色。

conicalGradient.setColorAt(0.0, Qt::red); 将 0 度角设置为红色；

conicalGradient.setColorAt(60.0/360.0, Qt::yellow);将 60 度角设置为黄色。

由于一个圆周是 360 度，所以 60.0/360.0 即是这个角度的比例。其余代码以此类推。最后一句，我们将 1.0 处设置为红色，也就是重新回到起始处。至于颜色的分布，这是由颜色空间定义的，有兴趣的朋友可以查阅有关颜色模型的理论。

painter.translate(r, r);

这是我们唯一不熟悉的函数。QPainter::translate(x, y) 函数意思是，将坐标系的原点设置到 (x, y) 点。原本坐标系原点位于左上角，我们使用 translate(r, r)，将坐标原点设置为 (r, r)。这么一来，左上角的点的坐标就应该是 (-r, -r)。

最后，我们使用 drawEllipse() 函数绘制圆盘。注意，由于我们已经把坐标原点设置为 (r, r)，因此，在绘制时，圆心应该是新的坐标 (0, 0)，而不是原来的 (r, r)。

**Qt ListWidget的方法使用功能：**

listWidget = QListWidget() #实例化一个(item base)的列表

listWidget.addItem('dd') #添加一个项

listWidget.addItems([]) # 从序列中添加子项

listWidget.setDragEnabled(True) #设置拖拉

listWidget.sortItems() #排序

listWidget.selectAll()全选

listWidget.setSortingEnabled(bool) #设置自动排序

listWidget.setSelectionMode(QtGui.QAbstractItemView.ExtendedSelection) #设置选择模式

选择模式有:ExtendedSelection 按住ctrl多选, SingleSelection 单选 MultiSelection 点击多选 ContiguousSelection 鼠标拖拉多选

listWidget.setCurrentRow(0) 设置当前选择行默认为-1

listWidget.count() #得到子项总数

listWidget.item(row).text() #得到第 row 行的内容 listWidget.item(row) 返回一个item 对象

listWidget.takeItem(row) #返回row 行的所在的item 对象 可以用在 insertItem（）中

listWidget.insertItem(2,item) #在第二行插入一项 item 可谓为一个listviewitem对象 或者string

listWidget.setCurrentItem('dd') #设置'dd'为当前项

listWidget.selectedItems() 返回一个包含item对象 的list 对象

修改item 的内容

item.setText('dsds') # 设置item的内容为dsds item为对象 可从 listWidget.item(row) takeItem(row) 得到

获取listWidget 的当前行行号

 connect(musicList,SIGNAL(currentRowChanged(int)),this,SLOT(musicListRowNum(int)));  //信号 与槽函数的链接

**去掉滚动条**

list->setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOff); //垂直方向

list->setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff); //水平方向

将qtListwight中的项转化成char \*

printf("%s\n",Seclist->item(SeclistFlag)->text().toLatin1().data());

**Qt控件抓屏Widget：**

**QPixmap pix = QPixmap::grabWidget(this,0,0,720,1280);**

**pix.save("pixture.png","png");**

路径\文件名字 文件类型