1 QList 类

//定义链表
1)QList<int> list;
2)
typedef sturct
{
    int id;
    char name[10];
}STU;
STU stu;
stu.id= 1;
strcpy(stu.name,”abc”);
QList<STU> listStu; //链表成员为结构
listStu.append(*stu);//增加一节点
printf(“id=%d\n”, listStu.at(0).id);//把链表中的节点信息打印出。
clistStu.clear();//清除链表内容
removeItem(int i)//删除第几个节点
listStu.insert(…….);
count()//链表节点数

2 QpushButton

setMaximumWidth(int)
setMaximumHeight(int)
setMinimumWidth(int)
setMinimumHeight(int)
setText(QStrng)
setIcon((QIcon("/mnt/hgfs/hard/fcw.ico")));//设置图标

3 QRadioButton

单选按钮，但同一窗口在不同 qgroupbox 内的均可先中一个。
例子：
radioA->setChecked(true);
if(radioA->isChecked())
        qDebug()<<"A is check";

例子：
```
QRadioButton *radSexA  = new QRadioButton(″男″);
```

```
QRadioButton *radSexB  = new QRadioButton("女");
QVBoxLayout *vlaySex = new QVBoxLayout;
vlaySex->addWidget(radSexA);
vlaySex->addWidget(radSexB);
QGroupBox *grpSex = new QGroupBox("sex");
grpSex->setLayout(vlaySex);
QRadioButton *radEduA  = new QRadioButton("小学");
radEduA->setChecked(true);
QRadioButton *radEduB  = new QRadioButton("大学");
if(radEduB->isChecked())
    qDebug()<<"selected";
QVBoxLayout *vlayEdu = new QVBoxLayout;
vlayEdu->addWidget(radEduA);
vlayEdu->addWidget(radEduB);
QGroupBox *grpEdu = new QGroupBox("edu");
grpEdu->setLayout(vlayEdu);
QHBoxLayout *hlayMain = new QHBoxLayout;
hlayMain->addWidget(grpSex);
hlayMain->addWidget(grpEdu);
this->setLayout(hlayMain);
```

4 QCheckBox

检查框，可多选。
setChecked(bool) 设置状态
checkState() 获取状态

例子代码段，供参考

```
chk1 = new QCheckBox("checkbox1");
QCheckBox *chk2 = new QCheckBox("checkbox2");
QCheckBox *chk3 = new QCheckBox("checkbox3");
QPushButton *btn = new QPushButton("check");

QVBoxLayout *layM = new QVBoxLayout;
layM->addWidget(chk1);
layM->addWidget(chk2);
layM->addWidget(chk3);
layM->addWidget(btn);
connect(btn, SIGNAL(clicked()), this, SLOT(slotSend()));
this->setLayout(layM);
this->resize(300,200);
```

5 QComboBox

currentText();//读取当前值
addItem(QString) //增加一个下拉内容
insertItem(int, QString);//插入一个下拉内容。
setMaxCount(int)//设置最大显示数，超过部分不显示。
setCurrentIndex(int);//设置显示的索引指定的值。
com->findText(QString);//查找指定内容的索引值。

例子：

```
QLabel *lblSex = new QLabel("性别:");
QComboBox *box  = new QComboBox;
box->addItem("男");
box->addItem("女");
box->insertItem(1, "未知");
box->setCurrentIndex(box->findText("未知"));
QHBoxLayout *hlayMain = new QHBoxLayout;
hlayMain->addWidget(lblSex);
hlayMain->addWidget(box);
this->setLayout(hlayMain);
```

6 QLineEdit

单行文本输入框。
setReadOnly(True) #设置为只读
setDragEnabled(True) #设置能接受拖放
setMaxLength(5) #设置最大长度
selectAll() #全选
setFocus() #得到焦点
setEchoMode(QLineEdit::Password);//设置显示密文
text()//

7 QTextEdit

toPlainText();//读取 **QTextEdit** 的值
moveCursor(QTextCursor::End);//光标移到最后,前提先要获得光标 setFocus()
setFocus();//获得光标
append(QString);//在末尾增加新的一行数据
textCursor().insertText(QString); //在光标处插入数据

8 QDateTimeEdit

setDateTime(QDatetime.currentDateTime());//设置时间
setCalendarPopup(true);//设置下拉选择日期。。。
setDisplayFormat("dd/M/yyyy");//显示格式
例子：

```
QDatetime dt;
QDatetimeEdit * dtEdit = new QDatetimeEdit;
dt = QDateTime::fromString("2012-08-19","yyyy-MM-dd");
//dt = QDateTime::currentDateTime();
dtEdit->setDateTime(dt);
dtEdit->setDateTime(QDateTime::fromString("2012-03-12","yyyy-MM-dd"));
```

9 QtableWidget

QTableWidget 是 QT 程序中常用的显示数据表格的空间，很类似于 VC、C#中的 DataGrid。说到 QTableWidget，就必须讲一下它跟 QTabelView 的区别了。QTableWidget 是 QTableView 的子类，主要的区别是 QTableView 可以使用自定义的数据模型来显示内容(也就是先要通过 setModel 来绑定数据源)，而 QTableWidget 则只能使用标准的数据模型，并且其单元格数据是 QTableWidgetItem 的对象来实现的(也就是不需要数据源，将逐个单元格内的信息填好即可)。这主要体现在 QTableView 类中有 setModel 成员函数，而到了 QTableWidget 类中，该成员函数变成了私有。使用 QTableWidget 就离不开 QTableWidgetItem。QTableWidgetItem 用来表示表格中的一个单元格，正个表格都需要用逐个单元格构建起来。
属性：

```
//初始行、列
tableWidget = new QTableWidget(0,3);
//设置表头
QStringList listHead ;
listHead<<"id"<<"name";
```

或

```
tableWidget->setHorizontalHeaderLabels(QStringList()<<tr("用户 id")<<tr("用户名")<<tr("用户密码")); //水平表头，垂直方向为 setVerticalHeaderLabels

tableWidget->setSelectionBehavior(QAbstractItemView::SelectRows);//选中一行，单个单元格为 SelectItems，选中一列 SelectColumns

//不可编辑
tableWidget->setEditTriggers(QAbstractItemView::NoEditTriggers);

selectAll();//选中所有记录

selectRow (int);//选中指定行， 0,1，2,3
```

setRowCount(int); //设置行数为 10

setColumnCount(int); //设置列数为 5

setWindowTitle(QString); //设置标题

resize(350, 200); //设置表格初始大小

rowCount();//获取总的行数,表头不算一行

columnCount();//获取总的列数

insertRow(int);//插入一空行，行数 0,1,2

setItem(int, int, QTableWidgetItem);//直接给一个单元格赋值。

QTableWidgetItem *columnHeaderItem0 = tableWidget->horizontalHeaderItem(0); //获得水平方向表头的 Item 对象

//取当前选中行记录值:
QTableWidget *tableWidget = new QTableWidget;
QTableWidgetItem *item = tableWidget->item(tableWidget->currentRow(),1);
QMessageBox::warning(this,"imfo",item->text());

取选中行的值(可多行)
QTableWidget table;
////打印出的值是按列，从上到下依次取。
```
for(i = 0 ; i < table->selectedItems().count(); i++)
    {
        cout<<"values="<<table->selectedItems().at(i)->text().toStdString().c_str()<<endl;
    }
```
常用信号：
itemSelectionChanged　//光标改变当前行。
itemClicked　//鼠标单击选中的行触发
setcolumnwidth(0,100);//设置列宽度

设置行颜色

```
//选中行颜色
tableWidget->setStyleSheet("selection-background-color: blue");
//行背景色
tableWidget->setStyleSheet("QTableView::Item{background-color:#FF3EFF}");
```

10 QtreeWidget

QTreeWidget * treeWidget = new QTreeWidget;
currentItem(); //当前选中的树节点
selectedItems().at(0);//选中节点的第几个
QTreeWidgetItem //树的节点类型
QTreeWidgetItem* parent=item->parent(); //找节点 item 的上一层节点。
setWindowTitle(QString); //设置表头
例子：

```
treeWidget->setHeaderLabels(QStringList()<<"Key"<<"Value");
```

或

```
QStringList ss;
ss<<"key"<<"values";
treeWidget->setHeaderLabels(ss);

//一级节点
QTreeWidgetItem *A = new QTreeWidgetItem(QStringList()<<"AAA"<<"AAA1");
A->setCheckState(0, Qt::Checked);//节点增加复选框,且为选中状态。Unchecked 为没选中
treeWidget->addTopLevelItem(A);//增加在一级

//二级节点
QStringList slistA;
slistA<<"a1"<<"aa2";
QTreeWidgetItem *sub1 = new QTreeWidgetItem(slistA);
sub1->setCheckState(0,Qt::Checked);
A->addChild(sub1);

expandAll//节点展开
节点.checkState();//判断节点是否选中
遍历树节点：
QTreeWidgetItemIterator it(treeWidget);
while (*it)
{
    //对(*it)进行处理，(*it)就是 QTreeWidgetItem *类型的，比如(*it)->text(0)
     ((*it)->checkState(0))
    qDebug()<< (*it)->text(1);
    ++it;
}//这个循环会对所有 item 进行遍历，方式为 先序遍历
```

例子：

```
#include "tree.h"
#include <iostream>
using namespace std;
```

```
TREE::TREE()
{
    treeWidget = new QTreeWidget;
    connect(treeWidget,     SIGNAL(itemChanged(QTreeWidgetItem*,int)),     this,
SLOT(treeItemChanged(QTreeWidgetItem*,int)));
    treeWidget->setWindowTitle("QTreeWidget");
    //设定头项名称
    //treeWidget->setHeaderLabels(QStringList()<<"Key"<<"Value");
    QStringList ss;
    ss<<"key"<<"values";
    treeWidget->setHeaderLabels(ss);
    //一级节点
    QTreeWidgetItem *A = new QTreeWidgetItem(QStringList()<<"AAA"<<"AAA1");
    A->setCheckState(0, Qt::Checked);
    treeWidget->addTopLevelItem(A);
    //二级节点 1
    QStringList slistA;
    slistA<<"a1"<<"aa2";
    QTreeWidgetItem *sub1 = new QTreeWidgetItem(slistA);
    sub1->setCheckState(0,Qt::Checked);
    A->addChild(sub1);

    //二级节点 2
    slistA.clear();
    slistA<<"a2"<<"aaad";
    QTreeWidgetItem *sub2 = new QTreeWidgetItem(slistA);
    sub2->setCheckState(0,Qt::Unchecked);
    A->addChild(sub2);
    QHBoxLayout *lay = new QHBoxLayout;
    lay->addWidget(treeWidget);
    this->setLayout(lay);
    QTreeWidgetItemIterator it(treeWidget);
    while (*it)
    {
        //对(*it)进行处理，(*it)就是 QTreeWidgetItem *类型的，比如(*it)->text(0)
        if((*it)->checkState(0) && (*it)->childCount() == 0)
                qDebug()<< (*it)->text(1)<< ((*it)->parent())->text(1);
        ++it;
    }//这个循环会对所有 item 进行遍历，方式为 先序遍历
}
void TREE::treeItemChanged(QTreeWidgetItem* item,int column)
{
    QString itemText=item->text(0);
    if(Qt::Checked==item->checkState(0))  //选中时
```

```
    {
        QTreeWidgetItem* parent= item->parent();
        int count=item->childCount();
        if(count>0)
        {
            for(int i=0;i<count;i++)
            {   //子结点也选中
                item->child(i)->setCheckState(0,Qt::Checked);
            }
        }
        else
        {   //是字节点
            updateParentItem(item);
        }
    }
    else if(Qt::Unchecked==item->checkState(0))
    {
        int count=item->childCount();
        if(count>0)
        {
            for(int i=0;i<count;i++)
            {
                item->child(i)->setCheckState(0,Qt::Unchecked);
            }
        }
        else
        {
            updateParentItem(item);
        }
    }
}


void TREE::updateParentItem(QTreeWidgetItem* item)
    {
        QTreeWidgetItem* parent=item->parent();
        if(parent==NULL)
        {
            return;
        }
        //选中的字节点个数
        int selectedCount=0;
        int childCount=parent->childCount();
        for(int i=0;i<childCount;i++)
```

```
        {
            QTreeWidgetItem *childItem=parent->child(i);
            if(childItem->checkState(0)==Qt::Checked)
            {
                selectedCount++;
            }
        }

        if(selectedCount<=0)
        {   //选中状态
            parent->setCheckState(0,Qt::Unchecked);
        }
        else if(selectedCount>0&&selectedCount<childCount)
        {   //部分选中状态
            parent->setCheckState(0,Qt::PartiallyChecked);
        }
        else if(selectedCount==childCount)
        {   //未选中状态
            parent->setCheckState(0,Qt::Checked);
        }

    }
```

例子：

tree.cpp.txt　tree.h.txt

11 类型转换

1)时间转字符串

```
dt.date().currentDate().toString("yyyyMMdd");
```

2)字符串转时间

```
QDateTimeEdit *dtEdit;
dtEdit->setDateTime(QDateTime::fromString("2012-03-12","yyyy-MM-dd"));
```

或

```
QDateTime dt;
dt = QDateTime::fromString("2012-08-19","yyyy-MM-dd");
dtEdit->setDateTime(dt);
```

12 工具栏

```
QToolBar *toolBar = new QToolBar;
```

```
toolBar->addAction(QIcon(tr("/home/yzj/work/qt/test/MSN.ICO")),tr("&File"));
this->addToolBar(toolBar);
```

13 状态栏

```
    QLabel *lblUserName = new QLabel;
QStatusBar statusBar = new QStatusBar;
    lblUserName->setMinimumSize(100,20);
    lblUserName->setFrameShadow(QFrame::Sunken);
    lblUserName->setFrameShape(QFrame::WinPanel);
    lblUserName->setText(tr("工号"));
    statusBar->addWidget(lblUserName);
    this->setStatusBar(statusBar);
```

14 工作区

```
QWorkspace workSpace = new QWorkspace();
this->setCentralWidget(workSpace);
```

15 菜单栏及菜单

```
   //创建存放菜单的 menubar
    QMenuBar *menuBar = new QMenuBar;

    //创建一级菜单 Edit
    QMenu *menuEdit = new QMenu("Edit");
    menuBar->addMenu(menuEdit);

    //二级菜单
    QAction *actionCut = menuEdit->addAction("cut");
    connect(actionCut, SIGNAL(triggered()), this, SLOT(slotTest()));

    menuEdit->addAction("copy");
    QMenu * menuSearch = menuEdit->addMenu("Search");

    //三级菜单
    QAction *actSearch = menuSearch->addAction("up search");
    menuSearch->addAction("down search");
    actSearch->setShortcut(tr("ctrl+s"));
    actSearch->setIcon(QIcon(tr("//mnt/hgfs/hard/fcw.ico")));
```

```
    this->setMenuBar(menuBar);
```

QWidget 类

```
SetWindowIcon(Icon()); //设置窗口图标
resize(…)
move(…)
```

Qt 中 int 转换成 QString

2012-04-01 19:18:51| 分类： QT 开发 | 标签： |字号大中小 订阅

有两种方法
1.使用

```
QString QString::number ( long n, int base = 10 )   [static]
```

如:

```
long a = 63;  QString s = QString::number(a, 10);      // s == "63"
QString t = QString::number(a, 16).toUpper();          // t == "3F"
```

加载背景:

```
    //加背景图
    QPalette p = palette();  //  得到窗口部件的调色板
    QPixmap img("/root/study1205/P2030468.JPG");  //蓝色图片
    p.setBrush(QPalette::Window, QBrush(img)); //给窗体设置笔刷，用笔刷设置
    setPalette(p);
    this->autoFillBackground();
    this->setWindowState(Qt::WindowMaximized);
```

重写画板事件,加到从 qwidget 或 qdialot 的子类中 ：

```
void TEST::paintEvent(QPaintEvent *e)
{
    QPainter painter(this);

painter.drawPixmap(0,0,this->width(),this->height(),QPixmap("/root/study1205/P2
030468.JPG"));
}
```