Code Example for SlidingStackedWidget class in Qt

Overview

This Article gives a full Qt Creator Code Example Project for the QStackedWidget class as described in the article Extending QStackedWidget for sliding page animations in Qt

This code example can be self-signed.

Postconditions

When successfully building and running this code example, the result should be as illustrated in the following clip:

The media player is loading...

Source files

Complete Source files and header files are part or the attached ZIP archive, that also includes the Qt Creator .pro file.

SlidingStackedWidget.cpp

```
#include "SlidingStackedWidget.h"


SlidingStackedWidget::SlidingStackedWidget(QWidget *parent)
    : QStackedWidget(parent)
{
    if (parent!=0) {
            m_mainwindow=parent;
        }
        else {
            m_mainwindow=this;
            qDebug()<<"ATTENTION: untested mainwindow case !";
        }
        //parent should not be 0; not tested for any other case yet !!
#ifdef Q_OS_SYMBIAN
#ifndef __S60_50__
        qDebug()<< "WARNING: ONLY TESTED AND 5TH EDITION";
#endif //__S60_50__
#endif //Q_OS_SYMBIAN
        //Now, initialize some private variables with default values
        m_vertical=false;
        //setVerticalMode(true);
        m_speed=500;
        m_animationtype = QEasingCurve::OutBack;  //check out the QEasingCurve documentation for different styles
        m_now=0;
        m_next=0;
        m_wrap=false;
        m_pnow=QPoint(0,0);
        m_active=false;
}


SlidingStackedWidget::~SlidingStackedWidget(){
}

void SlidingStackedWidget::setVerticalMode(bool vertical) {
        m_vertical=vertical;
}

void SlidingStackedWidget::setSpeed(int speed) {
        m_speed = speed;
}

void SlidingStackedWidget::setAnimation(enum QEasingCurve::Type animationtype) {
        m_animationtype = animationtype;
}

void SlidingStackedWidget::setWrap(bool wrap) {
        m_wrap=wrap;
}

void SlidingStackedWidget::slideInNext() {
        int now=currentIndex();
        if (m_wrap||(now<count()-1))
                // count is inherit from QStackedWidget
                slideInIdx(now+1);
}


void SlidingStackedWidget::slideInPrev() {
        int now=currentIndex();
        if (m_wrap||(now>0))
                slideInIdx(now-1);
}

void SlidingStackedWidget::slideInIdx(int idx, enum t_direction direction) {
        //int idx, t_direction direction=AUTOMATIC
        if (idx>count()-1) {
            direction=m_vertical ? TOP2BOTTOM : RIGHT2LEFT;
```


(C) Copyright Nokia 2013. All rights reserved.

```
            idx=(idx)%count();
        }
        else if (idx<0) {
            direction= m_vertical ? BOTTOM2TOP: LEFT2RIGHT;
            idx=(idx+count())%count();
        }
        slideInWgt(widget ( idx ),direction);
        //widget() is a function inherited from QStackedWidget
}


void SlidingStackedWidget::slideInWgt(QWidget * newwidget, enum t_direction direction) {

        if (m_active) {
                return; // at the moment, do not allow re-entrance before an animation is completed.
                //other possibility may be to finish the previous animation abrupt, or
                //to revert the previous animation with a counter animation, before going ahead
                //or to revert the previous animation abrupt
                //and all those only, if the newwidget is not the same as that of the previous running animation.
        }
        else m_active=true;

        enum t_direction directionhint;
        int now=currentIndex();  //currentIndex() is a function inherited from QStackedWidget
        int next=indexOf(newwidget);
        if (now==next) {
                m_active=false;
                return;
        }
        else if (now<next){
                directionhint=m_vertical ? TOP2BOTTOM : RIGHT2LEFT;
        }
        else {
                directionhint=m_vertical ? BOTTOM2TOP : LEFT2RIGHT;
        }
        if (direction == AUTOMATIC) {
                direction=directionhint;
        }
        //NOW....
        //calculate the shifts

        int offsetx=frameRect().width(); //inherited from mother
        int offsety=frameRect().height();//inherited from mother

        //the following is important, to ensure that the new widget
        //has correct geometry information when sliding in first time
        widget(next)->setGeometry ( 0,  0, offsetx, offsety );

        if (direction==BOTTOM2TOP)  {
                offsetx=0;
                offsety=-offsety;
        }
        else if (direction==TOP2BOTTOM) {
                offsetx=0;
                //offsety=offsety;
        }
        else if (direction==RIGHT2LEFT) {
                offsetx=-offsetx;
                offsety=0;
        }
        else if (direction==LEFT2RIGHT) {
                //offsetx=offsetx;
                offsety=0;
        }
        //re-position the next widget outside/aside of the display area
        QPoint pnext=widget(next)->pos();
        QPoint pnow=widget(now)->pos();
        m_pnow=pnow;

        widget(next)->move(pnext.x()-offsetx,pnext.y()-offsety);
        //make it visible/show
        widget(next)->show();
        widget(next)->raise();

        //animate both, the now and next widget to the side, using animation framework
        QPropertyAnimation *animnow = new QPropertyAnimation(widget(now), "pos");

        animnow->setDuration(m_speed);
        animnow->setEasingCurve(m_animationtype);
        animnow->setStartValue(QPoint(pnow.x(), pnow.y()));
        animnow->setEndValue(QPoint(offsetx+pnow.x(), offsety+pnow.y()));
        QPropertyAnimation *animnext = new QPropertyAnimation(widget(next), "pos");
        animnext->setDuration(m_speed);
        animnext->setEasingCurve(m_animationtype);
        animnext->setStartValue(QPoint(-offsetx+pnext.x(), offsety+pnext.y()));
        animnext->setEndValue(QPoint(pnext.x(), pnext.y()));

        QParallelAnimationGroup *animgroup = new QParallelAnimationGroup;

        animgroup->addAnimation(animnow);
        animgroup->addAnimation(animnext);

        QObject::connect(animgroup, SIGNAL(finished()),this,SLOT(animationDoneSlot()));
        m_next=next;
        m_now=now;
        m_active=true;
        animgroup->start();

        //note; the rest is done via a connect from the animation ready;
        //animation->finished() provides a signal when animation is done;
        //so we connect this to some post processing slot,
        //that we implement here below in animationDoneSlot.
}


void SlidingStackedWidget::animationDoneSlot(void) {
    //when ready, call the QStackedWidget slot setCurrentIndex(int)
    setCurrentIndex(m_next);  //this function is inherit from QStackedWidget
    //then hide the outshifted widget now, and  (may be done already implicitely by QStackedWidget)
    widget(m_now)->hide();
    //then set the position of the outshifted widget now back to its original
    widget(m_now)->move(m_pnow);
    //so that the application could also still call the QStackedWidget original functions/slots for changings
    //widget(m_now)->update();
    //setCurrentIndex(m_next);  //this function is inherit from QStackedWidget
    m_active=false;
    emit animationFinished();
}
```


(C) Copyright Nokia 2013. All rights reserved.

```
/* REFERENCES

http://doc.trolltech.com/4.6/animation-overview.html#easing-curves
http://doc.trolltech.com/4.6/qpropertyanimation.html
http://doc.trolltech.com/4.6/qanimationgroup.html

*/
```

SlidingStackedWidget.h

```
#ifndef SLIDINGSTACKEDWIDGET_H
#define SLIDINGSTACKEDWIDGET_H

#include <QStackedWidget>

#include <QtGui>
#include <QWidget>
#include <QDebug>
#include <QEasingCurve>

/*!
Description
        SlidingStackedWidget is a class that is derived from QtStackedWidget
        and allows smooth side shifting of widgets, in addition
        to the original hard switching from one to another as offered by
        QStackedWidget itself.
*/

class SlidingStackedWidget : public QStackedWidget
{
        Q_OBJECT

public:
        //! This enumeration is used to define the animation direction
        enum t_direction {
                LEFT2RIGHT,
                RIGHT2LEFT,
                TOP2BOTTOM,
                BOTTOM2TOP,
                AUTOMATIC
        };

        //! The Constructor and Destructor
        SlidingStackedWidget(QWidget *parent);
        ~SlidingStackedWidget(void);


public slots:
        //! Some basic settings API
        void setSpeed(int speed);   //animation duration in milliseconds
        void setAnimation(enum QEasingCurve::Type animationtype); //check out the QEasingCurve documentation for different styles
        void setVerticalMode(bool vertical=true);
        void setWrap(bool wrap);    //wrapping is related to slideInNext/Prev;it defines the behaviour when reaching last/first page

        //! The Animation / Page Change API
        void slideInNext();
        void slideInPrev();
        void slideInIdx(int idx, enum t_direction direction=AUTOMATIC);


signals:
        //! this is used for internal purposes in the class engine
        void animationFinished(void);

protected slots:
        //! this is used for internal purposes in the class engine
        void animationDoneSlot(void);

protected:
        //! this is used for internal purposes in the class engine
        void slideInWgt(QWidget * widget, enum t_direction direction=AUTOMATIC);

        QWidget *m_mainwindow;

        int m_speed;
        enum QEasingCurve::Type m_animationtype;
        bool m_vertical;
        int m_now;
        int m_next;
        bool m_wrap;
        QPoint m_pnow;
        bool m_active;

        QList<QWidget*> blockedPageList;
};

#endif // SLIDINGSTACKEDWIDGET_H
```

MainWindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QtGui/QMainWindow>

#include <QtGui>
#include <QPushButton>
#include <QCheckBox>
#include <QListWidget>
#include <QSlider>



#include "SlidingStackedWidget.h"

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = 0);
    ~MainWindow();
```


(C) Copyright Nokia 2013. All rights reserved.

```
protected:
    void createGuiControlComponents();
    void createMainLayout();
    void createSubSlidingWidgets();
    void createConnections();
    void createSlidingStackedWidget();
    QPushButton *buttonNext;
    QPushButton *buttonPrev;
    QCheckBox *checkWrap;
    QCheckBox *checkVertical;
    QSlider *slideSpeed;
    QComboBox *listAll;
    QLabel *speedLabel;
    QLCDNumber *speedDisplay;

    SlidingStackedWidget *slidingStacked;
    QVBoxLayout *mainLayout;
    QGridLayout *controlPaneLayout;
    QWidget *centralWidget;
    QWidget *slideWidget1;
    QWidget *slideWidget2;
    QWidget *slideWidget3;
    QWidget *slideWidget4;

    int animTime;
};

#endif // MAINWINDOW_H
```

MainWindow.cpp

```
#include "MainWindow.h"


// The Constructor calls the subfunctions for creation of the sample application
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
{
    animTime=500;
    createGuiControlComponents();
    createSubSlidingWidgets();
    createSlidingStackedWidget();
    createMainLayout();
    createConnections();
}

MainWindow::~MainWindow() {

}

void MainWindow::createGuiControlComponents() {
    int _min=500;
    int _max=1500;
    animTime=(_min+_max)>>1;

    buttonNext = new QPushButton(tr("Next"));
    buttonPrev = new QPushButton(tr("Prev"));
    checkWrap = new QCheckBox(tr("Wrap"));
    checkVertical = new QCheckBox(tr("Vertical"));

    listAll = new QComboBox();
    listAll->addItem(tr("Page 1"));
    listAll->addItem(tr("Page 2"));
    listAll->addItem(tr("Page 3"));
    listAll->addItem(tr("Page 4"));
    listAll->setMinimumHeight ( 40 );

    speedLabel = new QLabel(tr("Anim. Time:"));
    speedDisplay= new QLCDNumber();

    slideSpeed = new QSlider(Qt::Horizontal);
    slideSpeed->setMinimum(_min);
    slideSpeed->setMaximum(_max);

    //initialize slider and its display
    slideSpeed->setValue(animTime);
    speedDisplay->display(animTime);
}

void MainWindow::createMainLayout() {
    centralWidget=new QWidget(this);
    mainLayout=new QVBoxLayout();
    centralWidget->setLayout(mainLayout);
    controlPaneLayout=new QGridLayout();
    mainLayout->addWidget(slidingStacked);
    mainLayout->addLayout(controlPaneLayout);
    int row;
    row=1;
    controlPaneLayout->addWidget(buttonPrev,row,1,1,1);
    controlPaneLayout->addWidget(buttonNext,row,2,1,1);
    controlPaneLayout->addWidget(checkWrap,++row,1,1,1);
    controlPaneLayout->addWidget(checkVertical,row,2,1,1);
    controlPaneLayout->addWidget(speedLabel,++row,1,1,1);
    controlPaneLayout->addWidget(speedDisplay,row,2,1,1);
    controlPaneLayout->addWidget(slideSpeed,++row,1,1,2);
    controlPaneLayout->addWidget(listAll,++row,1,1,2);

    this->setCentralWidget(centralWidget);
}

void MainWindow::createSubSlidingWidgets() {
    slideWidget1=new QWidget();
    slideWidget2=new QWidget();
    slideWidget3=new QWidget();
    slideWidget4=new QWidget();
    QVBoxLayout *slideWidget1layout=new QVBoxLayout();
    slideWidget1->setLayout(slideWidget1layout);
    QVBoxLayout *slideWidget2layout=new QVBoxLayout();
    slideWidget2->setLayout(slideWidget2layout);
    QVBoxLayout *slideWidget3layout=new QVBoxLayout();
    slideWidget3->setLayout(slideWidget3layout);
    QVBoxLayout *slideWidget4layout=new QVBoxLayout();
    slideWidget4->setLayout(slideWidget4layout);
    QPushButton *b11=new QPushButton("Qt");
    slideWidget1layout->addWidget(b11);
    QPushButton *b12=new QPushButton("is cool !");
```


(C) Copyright Nokia 2013. All rights reserved.

```
    slideWidget1layout->addWidget(b12);

    QPushButton *b21=new QPushButton("Cool");
    slideWidget2layout->addWidget(b21);
    QPushButton *b22=new QPushButton("is Qt !");
    slideWidget2layout->addWidget(b22);

    QPushButton *b31=new QPushButton("Isn't");
    slideWidget3layout->addWidget(b31);
    QPushButton *b32=new QPushButton("Qt cool ?");
    slideWidget3layout->addWidget(b32);

    QPushButton *b41=new QPushButton("How cool");
    slideWidget4layout->addWidget(b41);
    QPushButton *b42=new QPushButton("is Qt !");
    slideWidget4layout->addWidget(b42);
}

void MainWindow::createSlidingStackedWidget() {
    slidingStacked= new SlidingStackedWidget(this);
    slidingStacked->addWidget(slideWidget1);
    slidingStacked->addWidget(slideWidget2);
    slidingStacked->addWidget(slideWidget3);
    slidingStacked->addWidget(slideWidget4);
    slidingStacked->setSpeed(animTime);
}

void MainWindow::createConnections() {
    QObject::connect(buttonNext,SIGNAL(pressed()),slidingStacked,SLOT(slideInNext()));
    QObject::connect(buttonPrev,SIGNAL(pressed()),slidingStacked,SLOT(slideInPrev()));
    QObject::connect(checkWrap,SIGNAL(clicked(bool)),slidingStacked,SLOT(setWrap(bool)));
    QObject::connect(checkVertical,SIGNAL(clicked(bool)),slidingStacked,SLOT(setVerticalMode(bool)));
    QObject::connect(slideSpeed,SIGNAL(valueChanged(int)),slidingStacked,SLOT(setSpeed(int)));
    QObject::connect(slideSpeed,SIGNAL(valueChanged(int)),speedDisplay,SLOT(display(int)));
    QObject::connect(listAll,SIGNAL(currentIndexChanged(int)),slidingStacked,SLOT(slideInIdx(int)));
}
```

'main.cpp

```
#include <QtGui/QApplication>
#include "MainWindow.h"

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;

#ifdef Q_OS_SYMBIAN
    w.showMaximized();
#else
    w.resize(360, 504);
    w.show();
#endif

    return a.exec();
}
```

SampleSlidingStackedWidget.pro

```
# -------------------------------------------------
# Project created by QtCreator 2010-03-21T15:45:08
# -------------------------------------------------
TARGET = SampleSlidingStackedWidget
TEMPLATE = app
SOURCES += main.cpp \
    MainWindow.cpp \
    SlidingStackedWidget.cpp
HEADERS += MainWindow.h \
    SlidingStackedWidget.h
```

Test application

Here the complete Sample Code Archive for download:

File:SampleSlidingStackedWidget.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.