Archived:Creating a loading animation with GIF, QMovie, and QLabel

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Qt Quick should be used for all UI development on mobile devices. The approach described in this article (using C++ for the Qt app UI) is deprecated.

Overview

This code example shows you how to create a simple component to use as an animated loading spinner with a GIF animation, `QMovie`, and `QLabel`.

You can find these GIF animation spinners with a simple Google search

Note: In order to use this code, you need to have Qt installed on your platform.

QAnimationLabel pro file

Let's start from the beginning, the project file. It is pretty much the normal project file but now there actually are resources which need to be included.

```
#  Copyright (c) 2009 Nokia Corporation
TEMPLATE = app
TARGET = AnimationLabel
QT += core \
    gui
HEADERS += qanimationlabel.h \
    animationlabel.h
SOURCES += qanimationlabel.cpp \
    main.cpp \
    animationlabel.cpp
FORMS +=
RESOURCES += animationlabel.qrc
```

qanimation.qrc

Here are the defined resources.

```
<!-- Copyright (c) 2009 Nokia Corporation  -->
<RCC>
    <qresource prefix="/" >
        <file>spinner-24x24.gif</file>
    </qresource>
</RCC>
```

QAnimationLabel usage

Header file

```
/*
 * Copyright (c) 2009 Nokia Corporation
 */

#ifndef ANIMATIONLABEL_H
#define ANIMATIONLABEL_H

#include <QtGui/QMainWindow>
#include <QtGui/QFrame>
#include <QtGui/QVBoxLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QPushButton>

#include "qanimationlabel.h"

class AnimationLabel : public QMainWindow
{
 Q_OBJECT

public:
 AnimationLabel(QWidget *parent = 0);
 ~AnimationLabel();

private:
};

#endif // ANIMATIONLABEL_H
```

Source file

(C) Copyright Nokia 2013. All rights reserved.

```
 /*
 * Copyright (c) 2009 Nokia Corporation
 */

#include "animationlabel.h"

AnimationLabel::AnimationLabel(QWidget *parent) : QMainWindow(parent) {
 /*
  * Spinner is loaded from the resources
  * defined in the animationlabel.qrc
  */
 QAnimationLabel* spinner = new QAnimationLabel(":/spinner-24x24.gif", this);

 /* Button to start the animation. */
 QPushButton* start = new QPushButton("Start");
 connect(start, SIGNAL(clicked()),
         spinner, SLOT(start()));

 /* Button to stop the animation. */
 QPushButton* stop = new QPushButton("Stop");
 connect(stop, SIGNAL(clicked()),
         spinner, SLOT(stop()));

 /* Layout for the UI elements. */
 QVBoxLayout* layout = new QVBoxLayout;

 /* Layout for the buttons. */
 QHBoxLayout* buttons = new QHBoxLayout;
 buttons->addWidget(start);
 buttons->addWidget(stop);

 /* Add spinner to the UI layout */
 layout->addWidget(spinner, 0, Qt::AlignCenter);
 /* Add buttons to the UI layout */
 layout->addLayout(buttons);

 /* Frame to be our container */
 QFrame* frame = new QFrame;
 /* UI layout as the containers layout */
 frame->setLayout(layout);
 /* Set frame as our central widget. */
 setCentralWidget(frame);
}

AnimationLabel::~AnimationLabel() {

}
```

QAnimationLabel

Header file

```
/*
 * Copyright (c) 2009 Nokia Corporation
 */

#ifndef QANIMATIONLABEL_H_
#define QANIMATIONLABEL_H_

#include <QVBoxLayout>

#include <QLabel>
#include <QMovie>

/**
 * QAnimationLabel
 *
 * Uses animation from the path
 * to display it in a QLabel.
 */
class QAnimationLabel : public QWidget {
 Q_OBJECT
public:
 QAnimationLabel(QString animationPath,
                 QWidget* parent);
 QAnimationLabel(QString animationPath,
                 QSize size,
                 QWidget* parent);
 virtual ~QAnimationLabel();

public slots:
 void start();
 void stop();

private:
 QPointer<QLabel> _container;
 QPointer<QMovie> _animation;

 void init(const QString& animationPath,
           const QSize& size);
};

#endif /* QANIMATIONLABEL_H_ */
```

Source file

```
/*
 * Copyright (c) 2009 Nokia Corporation
 */

#include "qanimationlabel.h"

QAnimationLabel::QAnimationLabel(QString animationPath,
                                 QWidget* parent) : QWidget(parent) {
 init(animationPath, QSize());
}
```


(C) Copyright Nokia 2013. All rights reserved.

```
QAnimationLabel::QAnimationLabel(QString animationPath,
                                 QSize size,
                                 QWidget* parent) : QWidget(parent) {
 init(animationPath, size);
}

QAnimationLabel::~QAnimationLabel() {
 _container->deleteLater();
 _animation->deleteLater();
}

void QAnimationLabel::init(const QString& animationPath,
                           const QSize& size) {
 /* We'll create the QMovie for the animation */
 _animation = new QMovie(animationPath);
 QSize s = size;
 /* If the size is empty we'll try to detect it.*/
 if(s.isEmpty()) {
  /* Go to the first frame.*/
  _animation->jumpToNextFrame();
  /* Take the size from there. */
  s = _animation->currentPixmap().size();
  /* Go back to the beginning. */
  _animation->jumpToFrame(0);
 }
 /* We're not automatically shown, so lets hide. */
 setHidden(true);
 /* We need a container for the QMovie, let's use QLabel */
 _container = new QLabel(this);
 /*
  * We'll set a fixed size to the QLabel
  * because we don't want to be resized
  */
 _container->setFixedSize(s);
 /* To get the movie displayed on the QLabel */
 _container->setMovie(_animation);
 /* To get the QLabel displayed we'll use a layout */
 QVBoxLayout* layout = new QVBoxLayout(this);
 /* Remove the all the extra space. */
 layout->setSpacing(0);
 layout->setMargin(0);
 /* Add the movie container to the layout */
 layout->addWidget(_container);
 /* Set the layout as our layout */
 setLayout(layout);
 /* Set our size fixed. */
 setFixedSize(s);
}


void QAnimationLabel::start() {
 /* Let's check if the movie can be started. */
 if(!this->_animation.isNull() &&
    (this->_animation->state() == QMovie::NotRunning ||
     this->_animation->state() == QMovie::Paused)) {
  /* It can so we'll start the animation... */
  this->_animation->start();
  /* ...and reveal ourself. */
  this->setHidden(false);
 }
}


void QAnimationLabel::stop() {
 /* Check if the animation can be stopped. */
 if(!this->_animation.isNull()) {
  if(this->_animation->state() == QMovie::Running) {
   /* It can so we'll stop the animation... */
   this->_animation->stop();
   /* ...and hide. */
   this->setHidden(true);
  }
 }
}
```

Postconditions

You know how to create a component to display a GIF animation with `QMovie` and `QLabel`.

Supplementary material

- You can test the AnimationLabel with a test application. The application is available for download at Media:AnimationLabel.zip.

See also

- QMovie documentation
- QLabel documentation

(C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.