Archived:Animating graphics item position in Qt

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 series of snippets demonstrates how to implement nice animation effects into your Qt UI application. Qt 4.6 has Qt Animation Framework as a built-in feature. Documentation can be found here: [1]

This snippet shows how to implement `QGraphicsItem` and animate a graphics item's position.

See full example code here: File:Animation fw.zip

Preconditions

Qt 4.6 has been installed.

Header file

The `QPropertyAnimation` class animates Qt properties. `QPropertyAnimation` interpolates over Qt properties.

```
#include <QtGui>

class PictureItem: public QObject, public QGraphicsPixmapItem
{
Q_OBJECT

    // Change picture position property for QPropertyAnimation
    Q_PROPERTY(QPointF pos READ pos WRITE setPos)

public:
    enum
    {
        PictureItemType = UserType + 1
    };

public:
    PictureItem(QSizeF size, const QPixmap& pixmap = 0, QObject* parent = 0);
    ~PictureItem();

    // Animate position of this class
    void animatePosition(const QPointF& start, const QPointF& end);

    QRectF boundingRect() const;

    int type() const
    {
        return PictureItemType;
    }

public slots:
    void animationFinished();

private:
    QSizeF m_size;
};
```

Source file

```
#include "mainwindow.h"

PictureItem::PictureItem(QSizeF size, const QPixmap& pixmap, QObject* parent) :
    QObject(parent), QGraphicsPixmapItem(pixmap)
{
    m_size = size;
    setCacheMode(DeviceCoordinateCache);
}

PictureItem::~PictureItem()
{
}
```


(C) Copyright Nokia 2013. All rights reserved.

```
QRectF PictureItem::boundingRect() const
{
return QRectF(0, 0, m_size.width(), m_size.height());
}

void PictureItem::animatePosition(const QPointF& start, const QPointF& end)
{
    // Start animate this class
    QPropertyAnimation* anim = new QPropertyAnimation(this, "pos");

    // 2 second duration animation
    anim->setDuration(2000);
    // position to start animation
    anim->setStartValue(start);
    // end position of animation
    anim->setEndValue(end);
    // easing curve
    anim->setEasingCurve(QEasingCurve::InOutElastic);

    // Listen animation finished signal
    QObject::connect(anim, SIGNAL(finished()), this, SLOT(animationFinished()));

    // Start animation and delete QPropertyAnimation class on the end
    anim->start(QAbstractAnimation::DeleteWhenStopped);
}

void PictureItem::animationFinished()
{
    // This slot is called when animation ends
}
```

Postconditions

`PictureItem` position can be animated by calling `PictureItem::animatePosition()`.

Animation Framework snippet series

Full example code package

- Enabling Qt Animation Framework in an application
- **Archived:Animating graphics item position in Qt**
- Archived:Animating graphics item rotation
- Archived:Animating graphics item fading
- Archived:Animating graphics item scaling
- Archived:Animating graphics item position and rotation simultaneously

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.