**Qt之设置QWidget背景色**

原创 2015年12月19日 11:32:28

* 20853
* 8
* 11

简述

QWidget是所有用户界面对象的基类，这意味着可以用同样的方法为其它子类控件改变背景颜色。

Qt中窗口背景的设置，下面介绍三种方法。

1. 使用QPalette
2. 使用Style Sheet
3. 绘图事件

一般我不用QSS设置窗口背景，也不建议使用。（这里是对于窗口而，如果是子部件当然可以）。因为窗口使用QSS设置背景之后，若子部件不使用同样的方式来设置，默认则会继承父窗口的样式。

* [简述](http://blog.csdn.net/liang19890820/article/details/50357715#简述)
* [使用QPalette](http://blog.csdn.net/liang19890820/article/details/50357715#使用qpalette)
* [使用Style Sheet](http://blog.csdn.net/liang19890820/article/details/50357715#使用style-sheet)
* [绘图事件](http://blog.csdn.net/liang19890820/article/details/50357715#绘图事件)
* [更多参考](http://blog.csdn.net/liang19890820/article/details/50357715#更多参考)

使用QPalette

使用QPalette来设置背景色

m\_pWidget = new QWidget(this); m\_pWidget->setGeometry(0, 0, 300, 100); QPalette pal(m\_pWidget->palette()); //设置背景黑色 pal.setColor(QPalette::Background, Qt::black); m\_pWidget->setAutoFillBackground(true); m\_pWidget->setPalette(pal); m\_pWidget->show();

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

使用Style Sheet

使用样式表来设置背景色，可以参考：[Qt Style Sheets文档](http://doc.qt.io/qt-5/stylesheet.html" \o "Qt Style Sheets文档" \t "_blank)

m\_pWidget = new QWidget(this); m\_pWidget->setGeometry(0, 0, 300, 100); m\_pWidget->setStyleSheet("background-color:black;"); m\_pWidget->show();

* 1
* 2
* 3
* 4

关于子类化QWidget，助手中有段说明：

// If you subclass from QWidget, you need to provide a paintEvent for your custom QWidget as below: void CustomWidget::paintEvent(QPaintEvent \*) { QStyleOption opt; opt.init(this); QPainter p(this); style()->drawPrimitive(QStyle::PE\_Widget, &opt, &p, this); }

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8

如果要子类化一个QWidget，为了能够使用样式表，则需要提供paintEvent事件。这是因为QWidget的paintEvent()是空的，而样式表要通过paint被绘制到窗口中。

**警告：**确保子类化的QWidget定义了Q\_OBJECT宏。

绘图事件

重写paintEvent，使用QPainter来绘制背景。

void Widget::paintEvent(QPaintEvent \*event) { Q\_UNUSED(event); QPainter p(this); p.setPen(Qt::NoPen); p.setBrush(Qt::black); p.drawRect(rect()); }

## Qt-QPalette类的用法

QPalette是Qt中的一个调色板类，用于美化界面。

### 1.构造函数

* QPalette(const QColor & button)
* QPalette(Qt::GlobalColor button)
* QPalette(const QColor & button, const QColor & window)
* QPalette(const QPalette & p)

常用的构造函数如上，一般用QColor进行初始化，button表示窗口控件颜色，window表示窗口颜色。   
QColor的初始化：

* QColor(int r, int g, int b, int a = 255)
* QColor(Qt::GlobalColor color)

其中Qt::GLobalColor包括：   
   
如Qt::green。

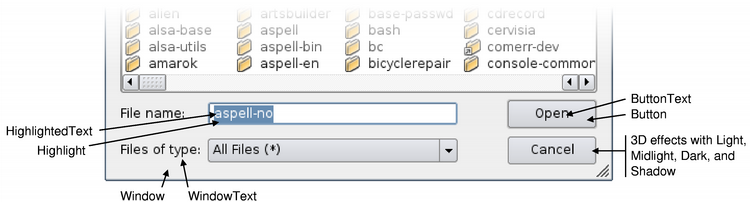
### 2.成员函数

（用于设置颜色）

* void setColor(ColorGroup group, ColorRole role, const QColor & color)
* void setColor(ColorRole role, const QColor & color)

### 3.ColorRole枚举项

| **ColorRole** | **作用域** |
| --- | --- |
| QPalette::Window | 窗口颜色 |
| QPalette::WindowText | 窗口文字颜色 |
| QPalette::Base | 例如TextEdit的背景色 |
| QPalette::Text | 例如TextEdit的文字色 |
| QPalette::Button | 按钮背景色 |
| QPalette::ButtonText | 按钮颜色 |
| QPalette::Hightlight | 高亮背景色 |
| QPalette::HightlightText | 高亮字体色 |
| QPalette::Link | 链接颜色 |
| QPalette::LinkVisited | 访问过的链接色 |



### 4.使用示范

(实现了一个编辑器颜色主题的变换）

//Qt 5.2.1 编译通过

//CopyRight by ChestnutHeng,2015

void MainWindow::on\_actionBlack\_triggered()

{

ui->actionBlack->setChecked(true);

ui->actionWhite->setChecked(false);

QPalette p = palette(); //声明

p.setColor(QPalette::Base,QColor(0,0,0));

p.setColor(QPalette::Text,QColor(180,180,180));

p.setColor(QPalette::HighlightedText,QColor(70,70,70));

p.setColor(QPalette::Highlight,QColor(200,200,200));

ui -> plainTextEdit -> setPalette(p); //使用

p.setColor(QPalette::Background,QColor(100,100,100));

this-> setPalette(p);

ui -> statusBar-> setPalette(p);

}