# Qt主题的颜色规范QPalette

## Qt主题如何在色彩上保持统一？

Qt5的基本控件全部由QStyle绘制，一般的Qt主题（QStyle）并没有关于颜色相关的配置文件，与之对应的，Qt提供了QPalette机制用于帮助控件及文字的绘制，主流的Qt主题一般都以QPalette提供的颜色为主体进行绘制，当然也并非不能用其它的颜色。 QPalette在主题中是很重要的一个部分，一般而言QStyle在绘制一个控件时会优先考虑使用QPalette中的颜色，这样就能够保证控件具有统一的风格。

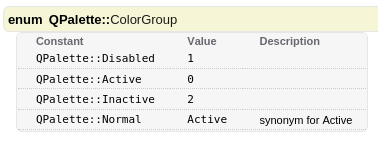


如果我们要开发一套自己的Qt主题，了解主流的规范是很有必要的。在社区中，这些规范往往环环相扣，从而形成了一个巨大的可以扩展的框架，虽然它们的学习成本可能比较高，但是当我们摸清楚后往往能够事半功倍。

## QPalette分析

### QPalette的三态

在QPalette中有QPalette::ColorGroup这一个枚举类型，它用来辅助主题获取当前控件的状态：



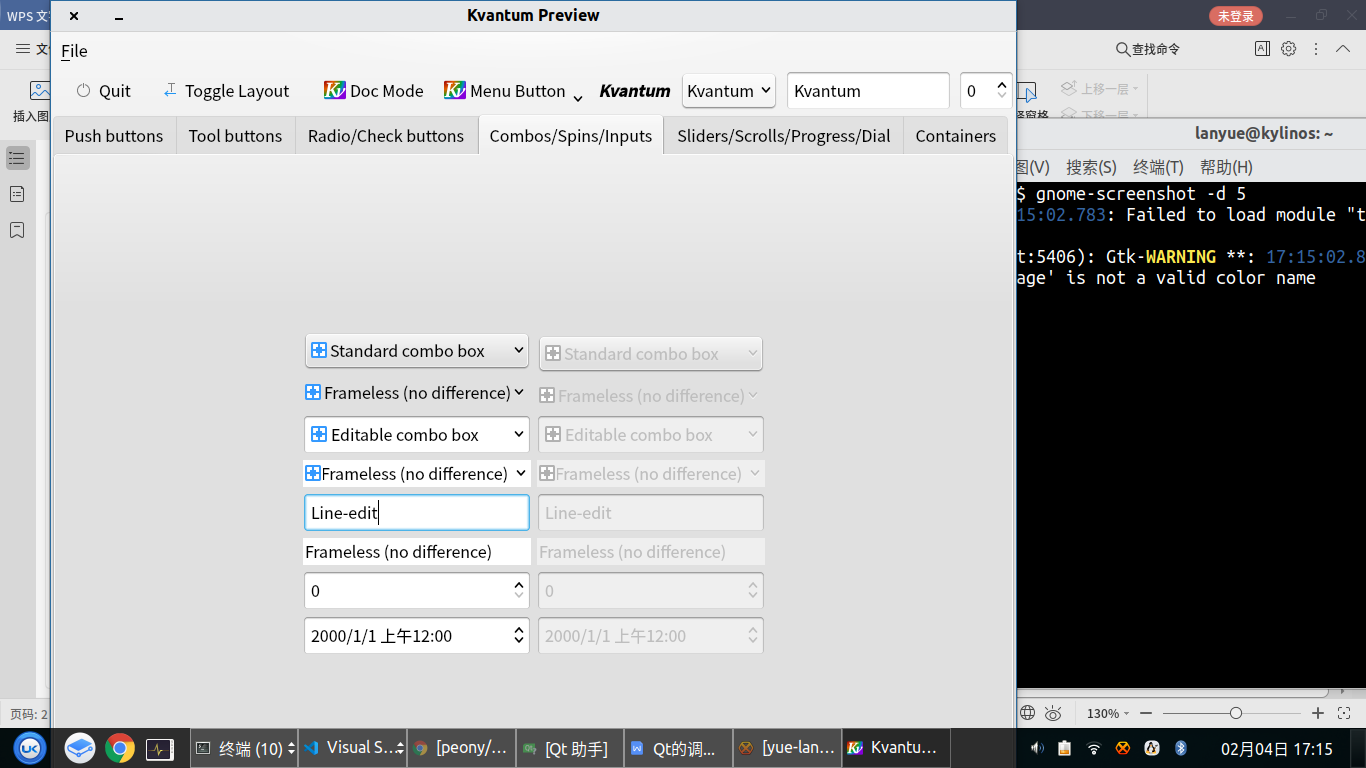
从官方的文档里面可以看出这个枚举有四个值，Disable、Active、Inactive和Normal，而Normal的值就是Active，所以QPalette总共只有三种状态。

Disable表示禁用状态，禁用情况下控件的颜色一般与普通状态控件的颜色区别较大，所有元素呈灰色；

Active则是控件激活的状态，一些激活的控件可能会有一些高亮的显示效果；

Inactive则是控件非激活时的状态，这里要与Disable区别开来，非激活的控件是可激活的，比如鼠标移动点击和键盘操作改变焦点至该控件，而Disable是无法激活的，也就是不能获取焦点。

一般来说，Active和Inactive的显示效果在一个主题内基本一致，如果有不同，往往也只是为了突出显示焦点控件。



我们能够从上图看到三种控件的状态，右侧控件的状态全部为Disable，左侧Line-edit控件则是具有焦点的Active状态，我们可以看到在这个主题(Oxygen)的设计中对于Active控件的文本框有一个边框高亮的效果，用来区分Inactive和Active状态。

对于不能够获取焦点的控件而言，理论上只有两种状态，Disable和非Disable，但这个时候根据控件不同，非Disable对应的状态枚举也可能不一样，有可能是Active也有可能是Inactive，这也是为什么Active和Inactive对应的颜色要接近的原因之一。

### QPalette中的颜色类型

颜色的类型在QPalette中对于的枚举是QPalette::ColorRole，我们同样参考Qt官方文档：

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| QPalette::Window | 10 | A general background color. |
| QPalette::Background | Window | This value is obsolete. Use Window instead. |
| QPalette::WindowText | 0 | A general foreground color. |
| QPalette::Foreground | WindowText | This value is obsolete. Use WindowText instead. |
| QPalette::Base | 9 | Used mostly as the background color for text entry widgets, but can also be used for other painting - such as the background of combobox drop down lists and toolbar handles. It is usually white or another light color. |
| QPalette::AlternateBase | 16 | Used as the alternate background color in views with alternating row colors (see [QAbstractItemView::setAlternatingRowColors](qthelp://org.qt-project.qtgui.5124/qtwidgets/qabstractitemview.html" \l "alternatingRowColors-prop)()). |
| QPalette::ToolTipBase | 18 | Used as the background color for [QToolTip](qthelp://org.qt-project.qtgui.5124/qtwidgets/qtooltip.html) and [QWhatsThis](qthelp://org.qt-project.qtgui.5124/qtwidgets/qwhatsthis.html). Tool tips use the Inactive color group of [QPalette](qthelp://org.qt-project.qtgui.5124/qtgui/qpalette.html), because tool tips are not active windows. |
| QPalette::ToolTipText | 19 | Used as the foreground color for [QToolTip](qthelp://org.qt-project.qtgui.5124/qtwidgets/qtooltip.html) and [QWhatsThis](qthelp://org.qt-project.qtgui.5124/qtwidgets/qwhatsthis.html). Tool tips use the Inactive color group of [QPalette](qthelp://org.qt-project.qtgui.5124/qtgui/qpalette.html), because tool tips are not active windows. |
| QPalette::PlaceholderText | 20 | Used as the placeholder color for various text input widgets. This enum value has been introduced in Qt 5.12 |
| QPalette::Text | 6 | The foreground color used with Base. This is usually the same as the WindowText, in which case it must provide good contrast with Windowand Base. |
| QPalette::Button | 1 | The general button background color. This background can be different from Window as some styles require a different background color for buttons. |
| QPalette::ButtonText | 8 | A foreground color used with the Button color. |
| QPalette::BrightText | 7 | A text color that is very different from WindowText, and contrasts well with e.g. Dark. Typically used for text that needs to be drawn where Text or WindowText would give poor contrast, such as on pressed push buttons. Note that text colors can be used for things other than just words; text colors are *usually* used for text, but it's quite common to use the text color roles for lines, icons, etc. |

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| QPalette::Light | 2 | Lighter than Button color. |
| QPalette::Midlight | 3 | Between Button and Light. |
| QPalette::Dark | 4 | Darker than Button. |
| QPalette::Mid | 5 | Between Button and Dark. |
| QPalette::Shadow | 11 | A very dark color. By default, the shadow color is [Qt::black](qthelp://org.qt-project.qtgui.5124/qtcore/qt.html" \l "GlobalColor-enum). |

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| QPalette::Highlight | 12 | A color to indicate a selected item or the current item. By default, the highlight color is [Qt::darkBlue](qthelp://org.qt-project.qtgui.5124/qtcore/qt.html" \l "GlobalColor-enum). |
| QPalette::HighlightedText | 13 | A text color that contrasts with Highlight. By default, the highlighted text color is [Qt::white](qthelp://org.qt-project.qtgui.5124/qtcore/qt.html" \l "GlobalColor-enum). |

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| QPalette::Link | 14 | A text color used for unvisited hyperlinks. By default, the link color is [Qt::blue](qthelp://org.qt-project.qtgui.5124/qtcore/qt.html" \l "GlobalColor-enum). |
| QPalette::LinkVisited | 15 | A text color used for already visited hyperlinks. By default, the linkvisited color is [Qt::magenta](qthelp://org.qt-project.qtgui.5124/qtcore/qt.html" \l "GlobalColor-enum). |

可以看到目前Qt5的调色板中有21种颜色，和三态结合，总共是63种颜色可以作用于主题，但是实际上我们在绘制控件的某一部分的时候往往是保持它的ColorRole不变，只改变它的状态，所以我们还是可以认为调色板中只有21种颜色，这种1对3的设计还是比较巧妙的。如果要开发一个新的主题，而这个新的主题需要能够支持切换控件的颜色，那么我们在设计的时候必须遵循调色板的规范，使用对应的调色板颜色为控件上色。

在我们的主题中，使用什么ColorRole对控件的某一部分上色是可以由我们决定的，但是由于QPalette的全局性，我们必须考虑在设计一个ColorRole的对应色彩时会不会对其它使用同样ColorRole的控件产生不利的影响。

### QPalette与主题的关系

由于Qt5已经将控件的绘制全部交由主题，实际上我们可以完全忽视QPalette，用写死的颜色去做一个Qt主题出来，但是这样我们也会失去颜色的可扩展性，这不是我们所乐意见到的。QPalette更像是一种指导和一种思路，用来引导设计师怎进行设计，开发人员进行开发。

### 针对特殊桌面应用

当我们主题中的标准调色板没办法满足设计师设计的控件时，我们需要采取一些特殊手段，对这些控件进行特殊的处理。

以开始菜单和任务栏为例，开始菜单和任务栏本身的颜色乃至样式都和标准控件的颜色相差很大，不适合使用主题中的颜色进行绘制，这就相当于我们需要使用另一个主题对开始其的控件进行绘制。

当然我们也可以通过一些手段将特殊的控件绘制整合到一个主题中，但我认为这违背了主题框架的设计原则，主题本来就是为了统一风格的，对于不统一的地方我们应该在应用级和控件级进行个性化的定制。需要注意，特殊控件对应主题的设计同样要遵循QPalette规范，就好比开始菜单和任务栏，它们虽然和普通应用的颜色和样式不同，但是内部控件的风格应该是一致的。

作为主题框架的开发者，我希望特殊的控件和应用越少越好，Qt的大部分控件以及其组合其实已经能够满足我们日常设计的80%（目前的情况比较特殊，因为我们优先开发了开始菜单和任务栏这些特殊应用，但是Linux桌面上更多的应用是像文件管理器，文本编辑器这样的应用），使用这些控件进行开发应该是我们的首选，如果和系统主题不符，我们也要优先考虑在主题层面上对控件进行修改，如果还是没有办法，那么我们只能自己造轮子去画一个控件出来，我们往往得花非常大的力气才能实现一个同时符合规范和设计需求的控件，这不是一个项目早期应该做的工作。

另外，我们的桌面环境主题难以统一的主要原因之一就是控件在设计风格时的不一致，这样会导致当我们用控制面板设置控件的样式时，我们需要做的处理也越多，达到的效果可能也越差，这对开发人员是很不利的，我们应该从全局的角度出发，尽量避免这类问题。

### 关于图标主题

我们上面所讲的主题是应用主题，通常意义上的主题除了应用主题之外还有图标主题，不幸的是目前的Linux桌面规范中图标主题是与应用主题是分割开来的，也就是说在一般的情况下，我们没办法通过修改主题而改变图标主题，这可能会带来一些问题，比如我们有一套黑色的图标主题，在白色应用主题的情况下比较美观，但是在黑色主题下效果就不太好了，解决这个办法最直接的方法是绘制黑白两套图标主题，与应用主题相对应。

当然这并不是说应用主题不能够对图标主题做任何事，由于Qt的标准控件由主题绘制，我们当然也可以对要绘制的图标进行预处理，但问题是，我们通常难以估计当我们这么做以后带来的影响。比如说我有一个白底黑图的按钮，它在点击之后变成黑底白图，我们可以考虑只通过主题的预处理，用一张图片展示两张图片的效果，然而如果我们的背景或者图片的基色不能确定，要实现理想的效果是不可能的，这时在主题中进行预处理的做法就弊大于利了，如果我们想这样做，可能必须要限定一个范围。

控件上图标与主题切换、状态响应的设计在UKUI3.0的设计稿中非常常见，然而目前我们还没能找到一个很好的解决办法，我认为我们可能需要从设计和开发两方出发，找到一个合理可行的方案。

## 制作一个符合规范的主题

### 从设计角度

一个规范必然会有它的局限性，如何在有限的条件下进行最合理和美观的设计，这是考验一个设计师的功底的活。一个好的主题不仅应该能在我们的自研应用中表现良好，还应该考虑兼顾社区的应用。

调色板只是设计需要遵循的规范之一，为了设计一个好的主题，设计师还应该要主动的了解Qt的控件集，这样才能够区分什么是普通，什么是特殊。即使不是为了设计主题，从项目的角度上来说，计师同样是一个项目的成员，为了完成一个项目，需要从设计上考虑实现的可行性，如果能够把设计效果和实际结合，并且划分为阶段，对项目的发展来说会更加有益。我们现在展现出来的是一个接近完成的设计稿，但是缺少了中途的小目标，有很多细节上的困难没有办法得到分解，不利于评估和后续工作的推进。

### 从代码实现

我们首先得了解Qt5控件基于主题的绘制框架，相关的官方和非官方的例子很多。作为程序员，学会搬砖是必须的，我们的工作都是从去看代码开始的，事实证明，我在这里讲了可能也没有用，毕竟talk is cheap。我认为对一个开发人员来说，我们至少要花30%以上的时间在研究代码上面，没有投入就没有产出，一味的coding不会提高生产力。我们现有的技术体系满足不了新UKUI3.0的需求，想要更近一步我们就得去学习新的知识和技术。

主题可以衍生出很多交叉学科，比如和窗口管理器结合（毛玻璃效果等），和动画结合等等，从Gtk路线迁移过来到现在，我们仍然还没有完全发掘Qt的潜力，在我看到了它的冰山一角之后，我希望能够建立一套完整的知识体系，作为UKUI3.0的开发指导，这样我们的技术才能够上升一个台阶，这对于公司和开发人员无疑是都有好处的。

## 写在最后

我希望有公司能够有更多的投入，将主题研发这一块作为对未来的投资。我在开发文件管理器的时候，深刻的感觉到我们目前在桌面这一块的技术积累过于停留在表层，方向过于分散，如果仍然以目前单个产品优先的方向前进下去，为了绩效，我们肯定没有办法往深了去钻，在桌面这一块的核心竞争力就体现不出来，这样形成不了一个良性的循环，为了更加贴近我们的理想，主题是很重要的基石。我们是终端产品中心，确实出发点和落脚点都是产品，但产品不等于单一应用，如果把整个UKUI算作产品，我们要顾及整个桌面，顾及它的生态等等，这些甚至比我们现在的工作更加重要，而且更加困难。从现在的开发路线来看，我认为我们已经产生偏离了，也许有各种不得不为之的原因，但是我们需要时刻提醒自己，我们正处在一个瓶颈期，我们在开发的时候应该考虑的更加长远一些，因为要突破这些瓶颈，光是埋头苦干是不够的。