# 新UKUI3.0关于主题部分的构想——Qt部分

## 背景

在19.10-20.04期间，我提出了基于qt5-gtk2-platformtheme构建ukui桌面环境的统一主题这一思路，使得qt和gtk的应用能够以一种较为完备的方式实现主题风格的统一。

然而，随着UKUI3.0新设计稿的出炉，我感觉qt5-gtk2-platformtheme已经无法满足设计的需求——比如自定义控件调色板，窗口的特效和动画等需求，gtk平台插件和qt内置的qss主题引擎兼容性很差等问题，目前的技术手段不能够很好的解决它们，因此我不得不寻求新的解决思路。

作为开发者，我希望关于主题和设计的问题80%以上能够在非应用层面解决，这样能够使得开发人员的生产力得到解放，从而能够更加专注于逻辑和功能的实现。qt5-gtk2-platformtheme的实现给我了一点启发——我认为从QPA层面解决大部分的设计需求是目前实现ukui3.0实现的最佳途径。

## 前言

在讲述具体的代码设计之前，我首先得解释一下为什么重新把主题又拿出来讲一遍。我们知道同一个桌面环境的应用不论是从设计还是实现上，都理应具有许多共性，但是每一个桌面环境的设计都是不同的，这些不同的“共性”其实上就是主题的由来。

虽然主题是“老调重弹”，但是我们需要抛开一些传统的认知，对于qt5来说，主题并不是一些简单的配置文件和图标的集合，这与gtk现在的做法和形式是完全不同的（虽然gtk在gtk2时代有过基于cairo的一个主题引擎，能够支持毛玻璃特效，但是到gtk3来看，似乎除了gtk内置的主题引擎之外，所有相关的项目都已经没有音讯了，当然这不是重点），qt5的主题本质上是一个接口的实现，这个接口是由qt定义的，然而它并没有像gtk一样把规范定死，所以就诞生了比gtk更加复杂的，各式各样、五花八门的主题。

回到正题，如何实现这些不同的共性呢？怎样实现这些共性才能算是好的实现方式呢？这确实让我思考了很长一段时间。

我认为，既然称之为共性，那么它必须是兼容的。首先，对于一个桌面环境而言，必定有桌面套件之外的应用，那么这些应用在桌面环境中与套件的融合度就是检测我们主题的兼容性的重要标准。我们知道deepin维护了一套dtk，具有一系列精美的控件，这些控件是具有共性的，然而如果将其放在deepin外的桌面环境，或者与qt原生应用进行对比，就显得比较突兀了。在我看来，这并不能说是一个成功的尝试。显然deepin也意识到了这一点，并且思考了其它手段弥补dtk的缺陷，但我认为deepin在这一块的做法我认为并不能够说是好的，所以我在参考的时候只取其形，不取其意。

要解决共性的问题，有了deepin的“前车之鉴”，我认为我们不能从dtk相同的层面入手，至少不是将其作为主要的层面，而应该像qt5-gtk2-platformtheme所做的一样，首先保证所有的qt应用具有统一的风格，这就不得不去研究Qt5的平台插件机制，摸清它的实现思路以及上下限。

## Qt5平台主题

平台主题（QPA PLatformTheme）是Qt5中引入的新概念之一，它通过在Qt应用初始化时加载平台插件的主题（style）实现所有Qt应用的风格统一（platformtheme插件本身类似于创建主题的工厂），用于取代Qt4中的style intergration。我还没有仔细研究过二者的不同，不过在我看来他们的目的都是为了统一qt应用的风格，而且同样是基于QStyle的，但既然qt4已经成为过去式，那么我们只需要考虑研究Qt5的平台插件实现即可。

Qt的Style机制功能之强，扩展性之好，可以用巧夺天工来形容，而跟QPA平台插件的结合使他更上一层楼，qt5-gtk2-platformtheme+peony-qt就是一个目前可以在ukui下产生显著效果的例子。这是如何实现的呢？在Qt5平台插件框架的设计中，有一个名为QT\_QPA\_PLATFORMTHEME的环境变量，就好像ukui中使用gsettings设置gtk主题一样，通过QT\_QPA\_PLATFORMTHEME，我们能够实现qt应用默认主题的切换。

目前社区中比较常见的qpa主题插件，除了gtk2，还有kde、breeze、kvantum、qt5ct等，我们通过这些主流的主题插件，能够对platformtheme有一定的了解。在这些插件中，我们发现它们虽然同样是qt的平台插件，但却存在包涵的关系，比如kde和breeze、kvantum，breeze和kvantum本身是kde的一个主题，但在kde的控制面板中，我们可以设置使用breeze还是kvantum（而kde本身并不出现在其中，其实是被隐藏了），这就是由“kde”实现的。对应的，qt5ct是为了在kde之外的桌面环境中实现对qt主题的管理。

之所以能够实现这一点，是因为Qt支持使用代理style，即QProxyStyle，这实际上是一种类似于类的继承的实现（注意在继承情况下QT\_QPA\_PLATFORMTHEME的值是不会改变的），通过这一点，kde和qt5ct可以继承自breeze或者其它的style，从而实现管理、修改、扩展等操作。当然，这样做的缺点也是有的，首先，这种分层次的主题框架，和我们所熟知的gtk主题框架具有很大的差异，由于qt style的自由和可扩展性，我们很难像gtk一样对其形成一个既定的规范，这加大了学习和构建的难度；其次，由于平台主题的选择受到环境变量的控制，这是一个非常不稳定的因素，如果有人对环境变量进行修改，可能会导致一系列的问题；另外，我们不排除style之间相互冲突的问题，最明显的问题就是kde和qt5ct不能共存的问题（在debian上有报告过这个bug，解决的方案是qt5ct在kde上不修改环境变量，即保持kde中的QPA平台主题环境变量为kde不变）。

## 主题插件应用范例

我们在上面把主题插件分为了2类，现在我们在这两类中各取一个作为范例来进行研究。

### 专注特效的主题——KVantum

KVantum是基于svg的主题引擎，使用svg贴图进行控件的绘制，同时，它提供了基于KWindowEffect的窗口特效（属于KF5，但需要KWin的支持），比如毛玻璃效果以及动画的效果。



图 1：当我们使用kde的控制面板systemsettings5将主题引擎切换到KVantum，就能够通过KVantum的控制面板KVantummanager对现有系统主题的样式进行修改，上图是kvglass+半透明&毛玻璃特效的示例图。

KVantum实现的效果无疑是强大的，但是并非没有缺陷。首先，KVantum基于svg贴图对控件进行绘制，这样做的话我们无法通过qt提供的调色板对控件进行颜色的修改，对于控件的样式，比如边缘弧度等，可能也不能够通过设置qss的方式进行调整；其次KVantum的透明和毛玻璃效果是面向全局的，不可以精确的设置某一个具体控件是否启用，动画的效果也类似。

### 管理主题的主题——qt5ct

Qt5ct（qt5 configure tool)旨在提供主题引擎的切换，同时提供基本样式、颜色和图标主题的管理（在非kde桌面环境中）。在ukui上，我们能够通过qt5ct提供的设置界面可以轻易的在不同的qt主题中进行切换，同时可以对其中的一些主题做出额外的调整。

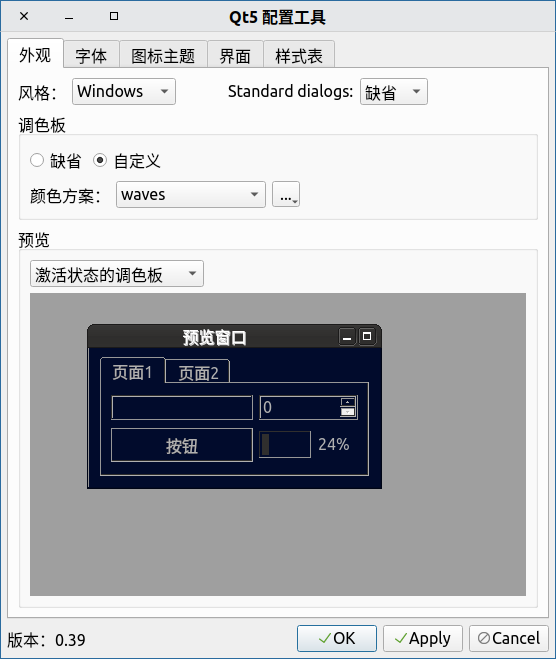


图 2：通过qt5ct我们可以对qt的主题进行进一步的调整，包括图标、字体、控件颜色、样式表、平台对话框等。然而qt5ct的缺陷固然是存在的，首先，就像之前所说，qt5ct提供的调色板和样式设置，无法与类似KVantum的主题兼容，因为KVantum本身并不支持控件的调色板，对于qt5ct来说，它无法辨别与其它主题的兼容性；其次，qt5ct的设置界面实在不能说的上优美，而且主题切换响应的速度也略慢，小问题也很多，虽然想法和思路很棒，但还不能说是一个很友好的项目。

## 再提kylin-style

记得我刚刚入职的时候，支撑组曾经提出过构建自己的qt系统主题，希望将其作为kylin-sdk的一部分，但由于各种因素没有推行下去，我现在提出的设想实际上是原来工作的一个强化版本。我希望能够通过新的平台插件，提供诸如毛玻璃、动画窗口等特效的实现和自定义设置的主题，同时使它能够兼容类似qt5ct所提供的调色板和样式表单定制功能，有了这样一个qt5主题引擎，就能够一次性解决我们目前应用风格和特效难以统一的问题，同时还能够避免dtk碰到的大坑。

从本质上来说，主题其实并不能被归为sdk，而是qt5的扩展。在本文中，能够勉强称得上sdk的可能就只有deepin的dtk。但是抛开sdk与否的问题来看，主题插件的做法确实更加先进，也许在之后的qt6会有更加完美的解决方案（比如原生支持css3.0，支持使用配置文件修改主题），但目前来看没有比基于qt5 qpa进行主题扩展的方式更加合理的方案了。

## qt5-style-ukui设计的总体框架和相关实现的思路

在调研了qt的主题框架之后，我计划实现一套主题框架，作为ukui今后应用开发的规范，暂时将其命名为qt5-style-ukui。

### 指导思路

1. 基于qt5ct的设计框架和思路，构建管理qt主题插件的主题插件qt5-style-ukui，并提供调色板和自定义qss的支持，在控制面板中提供设置接口
2. 基于breeze和breeze-dark实现可以自定义窗口特效qt5-style-ukui-white和qt5-style-ukui-black，并且与qt5-style-ukui提供的设置兼容，同样的，在控制面板中提供修改特效的接口
3. 可以通过配置文件对主题进行配置，支持qss和颜色、窗口特效的设置等

第一点能够满足设计稿中大部分的需求；第二点则是从我的角度出发，我希望窗口特效可以在控制面板中得到统一并且精确到控件类名的设置，而不仅仅是提供一个特效的开关；第三点是我考虑到设计与开发的协作，如果框架足够完善，我们就可以把大部分主题相关的工作交付给前端或者设计师，使得产品的效果更加贴近设计稿。

### 针对具体问题的实现思路

#### 关于窗口特效的实现

我参考了breeze、KVantum关于窗口特效的实现，关于毛玻璃的实现其实上借助了KWin，在KWindowEffect（KF5 API）中提供了调用KWin实现毛玻璃效果的接口，动画的话主要还是用到了Qt的Animation 相关的API，可能也有KWindowEffect的相关API。

我们需要劫持窗口的事件来触发特效的处理，具体的做法是通过重写QStyle的polish()函数（关于polish的介绍可以查看qt5助手），使用eventFilter机制实现窗口事件的劫持，对具体的事件进行处理，当然对应的在unpolish中同样需要进行处理，在劫持事件的过程中，我们可以通过QWidget::inhertes()对widget进行进一步的过滤，这也是自定义特效实现的关键。

#### 关于可扩展性和兼容性的实现

可扩展和兼容主要是针对qt5内置的调色板和qss机制而言的。正如我在KVantum的缺陷中所指出的，这个问题确实存在，而且我并没有完全摸清怎样实现才能够解决可扩展和兼容的问题，但是所幸我们在测试qt5ct+breeze时验证了breeze是基本上具有这个特性的，那么我们新的ukui-white/black的规范也可以基本定下来了

## 潜在的问题

### 关于窗口管理器

毛玻璃特效需要KWin的支持，之前我也与支撑组交流过关于ukwm向KWin迁移的问题，得到的回应也是积极的，只是其中必然还要牵扯到一些迁移不得不考虑的问题，比如我们现有的成果，像ukui-window-switch的后续工作，我了解到目前Kydroid2.0与现有的窗管进行了一些适配，虽然说Kydroid3.0可能会参考anbox的思路，抛弃现有双桌面的设计，但是类似这样的问题应该也是存在的，当然向KWin的迁移未必全是坏事，比如设计组设计的工作区间窗口拖拽就是KWin自带的功能，这一块的利与弊可能需要做一次比较完善的评估。

### 关于动画

说实话我还没怎么仔细研究过qt的动画框架（Animation），不知道它的上下限究竟何在，但我知道目前KDE确实有一些动画特效，并且能够进行相应的设置，我认为这一块需要有人能够去研究一下，比如在tab页中的切换的过渡效果是如何实现的（也许不是由主题控制，我并不清楚这一块），如果我们能够把动画特效做在主题插件中，对qt应用的开发来说将是很便利和友好的。

### 关于gtk应用

目前，我使用qt5-gtk2-platformtheme作为qt5的平台主题插件，实现了qt与gtk应用的风格统一，然而如果需要使用qt5-style-ukui作为新的平台主题插件，gtk与qt主题统一的这个问题我们就需要重新考虑了。这可能需要参考KDE或者lxqt的做法，但由于qt主题与gtk主题的差别巨大，做到统一主题是很难的。KDE和lxqt虽然能够支持gtk主题的设置，但是并没有提供统一设置qt和gtk主题的方法，这一块我目前还没有什么特别好的头绪。

### 关于Qt&KF5版本

目前的qt版本是5.12，而我们规定的标准版本是5.9，我在KVantum中发现其在5.11前后对毛玻璃特效的处理代码不同，主要是因为KF5在5.11提供了接口；然后还有一些关于dpr（设备像素比）相关的代码也受到Qt和KF5版本的影响，如果考虑迁移问题，这些问题也必须全部梳理一遍。

### 关于图标主题

图标主题和颜色主题的关系也是一个问题，deepin似乎对这个问题进行了处理，这一块我还没有太好的思路，我希望能够通过基于Qt的QIcon可扩展框架QIconEngine入手解决这个问题，但是我还没深入的研究过它。从技术上来说，它属于QPA之外的范畴，并不在本文的重点之内，但是从主题风格的统一上来说，这也是很关键的一点，其实我更希望能够有一套黑白通吃的万金油图标主题，这样就不用为这个问题而头痛了。