

Powerline 字体补丁

runsisi@hust.edu.cn @2013/05/03

接上一篇。

使用 spf13 的 vim 插件，如果当前系统的 vim 支持 python 的话，会安装 powerline 插件，如果不支持的话则会安装 vim-powerline，powerline 相当于 vim-powerline 的升级版，后者的效果可以参看上一篇，前者的效果可以看一下图：

```
24
25 " Setup Bundle Support {
26     " The next three lines ensure that the ~/.vim/bundle/ system work
27     filetype on
28     filetype off
29     set rtp+=~/.vim/bundle/vundle
30     call vundle#rc()
31 }
```

NORMAL 3.0 > ./vimrc

看的出 powerline 比 vim-power 的效果要炫的多，但是使用 powerline 的话需要对终端使用的字体打补丁，不然会显示乱码（当然乱码的原因也可能是终端编码设置不对，这里只考虑字体的原因）。

使用 gnome-terminal、konsole 等 linux 原生的终端也可以不给使用的字体打补丁，具体可以参考官方的文档，但如果使用 putty、securecr 等远程终端工具的话必须打补丁。

Powerline 插件自带了补丁工具，如果直接使用 spf13 的脚本安装的 powerline 插件，该工具在 spf13-vim/.vim/bundle/powerline/font 文件夹下：fontpatcher.py

使用类似下面的命令，就可以给 consola.ttf 字体打补丁：

```
./fontpatcher.py consola.ttf
```

我在使用该工具对 consolas 字体打补丁时遇到了一些问题：

1. The required FontForge modules could not be loaded.

该工具依赖 fontforge，但使用系统的包管理工具安装了 fontforge 之后可能仍然出现该问题（至少在我的 opensuse12.2 64bit 下不能用），可以通过下载源码重新编译（configure 时注意加上 --enable-pyextension 选项）；

2. 虽然重新编译、安装了 fontforge，执行该工具时仍然可能出现 1 中的问题

切换到 fontforge 源代码目录下的 pyhook 目录，然后执行 python setup.py install 安装 fontforge 的 python 模块（不知道为什么 fontforge 的 makefile 会缺这一步）

3. 打了补丁的字体 putty、securecr 等工具不认识，在字体设置无法选择打了补丁的字体，或者使用了打了补丁的字体后 powerline 插件显示的仍然是乱码

Github 上有打了补丁的 consolas 等字体，但都有这个问题，可能只适用于 linux 下的原生终端使用，在网上找到了一个该字体工具的补丁，但是是针对老版的 vim-powerline 的，和新版工具差异太大，已经不能用了，参考它的思路，对新版的 powerline 字体工具打了个补丁如下：

```
--- fontpatcher/old/fontpatcher.py 2013-05-03 23:23:11.788069924 +0800
```

```
+++ fontpatcher/new/fontpatcher.py 2013-05-03 23:23:53.145073351 +0800
```

```

@@ -22,14 +22,16 @@
    parser.add_argument('target_fonts', help='font files to patch', metavar='font', nargs='+',
type=argparse.FileType('rb'))
    parser.add_argument('--no-rename', help='don\'t add " for Powerline" to the font name',
default=True, action='store_false', dest='rename_font')
    parser.add_argument('--source-font', help='source symbol font', metavar='font',
dest='source_font', default='{0}/fontpatcher-symbols.sfd'.format(sys.path[0]),
type=argparse.FileType('rb'))
    +parser.add_argument('--fix-mono', help='fixes some mono-fonts which have glyphs of 0
widths', default=False, action='store_true', dest='fix_mono')
    args = parser.parse_args()

```

```

class FontPatcher(object):
-   def __init__(self, source_font, target_fonts, rename_font=True):
+   def __init__(self, source_font, target_fonts, rename_font=True, fix_mono=False):
        self.source_font = fontforge.open(source_font.name)
        self.target_fonts = (fontforge.open(target_font.name) for target_font in
target_fonts)
        self.rename_font = rename_font
+       self.fix_mono = fix_mono

    def patch(self):
        for target_font in self.target_fonts:
@@ -40,10 +42,10 @@

```

```

        # Rename font
        if self.rename_font:
-           target_font.familyname += ' for Powerline'
-           target_font.fullname += ' for Powerline'
+           target_font.familyname = 'Powerline ' + target_font.familyname
+           target_font.fullname = 'Powerline ' + target_font.fullname
            fontname, style = re.match("^[^-]*"?:(-.*))?$",
target_font.fontname).groups()
-           target_font.fontname = fontname + 'ForPowerline'
+           target_font.fontname = 'Powerline ' + fontname
            if style is not None:
                target_font.fontname += style
            target_font.appendSFNTName('English (US)', 'Preferred Family',
target_font.familyname)
@@ -102,6 +104,11 @@

```

```

        # Reset the font's glyph width so it's still considered monospaced
        target_font[source_glyph.unicode].width = target_font_width

```

```

+         if self.fix_mono:
+             for target_glyph in target_font.glyphs():
+                 if target_glyph.width == 0:
+                     target_glyph.width = target_font_width
+
+         target_font.em = target_font_em_original

        # Generate patched font
@@ -111,5 +118,5 @@
        extension = '.otf'
        target_font.generate('{0}{1}'.format(target_font.fullname, extension))

-fp = FontPatcher(args.source_font, args.target_fonts, args.rename_font)
+fp = FontPatcher(args.source_font, args.target_fonts, args.rename_font, args.fix_mono)
fp.patch()

```

使用如下的命令对 consola 字体打补丁，然后安装字体就可以在 putty、securecr 等工具中使用了：

```
./fontpatcher.py -fix-mono consola.ttf
```

我在 github 上传了一份打了补丁的 consolas 字体，下载之后可以直接使用：

<https://github.com/runsisi/consolas-font-for-powerline>

同时我也在 github 上上传了一份打包好了的 spf13 插件，只需要解压然后执行 ./install.sh 即可（注意要求 vim7.3，且 vim 支持 python）：

<https://github.com/runsisi/a-ready-to-use-vimrc>