## 为何选择Sublime Text?

我以前一直是[TextMate](https://github.com/textmate/textmate)的忠实用户。这是一个轻量级的、开源的软件，作为OS X的本地应用，具有很好的Mac风格。不过，虽然TextMate是一个很棒的编辑器，有时候还是显得功能不够。

我用过一些功能更强的软件，例如 [加上Python插件](http://confluence.jetbrains.com/display/PYH/PyCharm+IDE+and+Python+Plugin+for+IntelliJ+IDEA)的IntelliJ IDEA。 我特别喜欢它的debugger和test runner。不过，一个像IntelliJ这样的全功能的IDE对于中小项目来说还是显得过于庞大。

最近几周我开始越来越多的使用 [Sublime Text](http://www.sublimetext.com)。当我将它安装好之后，感觉非常不错。它确实非常快，自动定期的更新，以及更棒的是完全支持跨平台。对我来说，它最终胜过TextMate的地方是Sublime强大的插件子系统。对于Python开发，有不少插件可以让你开发起来更流畅、更有乐趣。

我现在仍然在不同的项目之间切换编辑器。不过我发现对应Python开发，Sublime在轻量级的编辑器和全功能的IDE之间有着很好的平衡。

## 字体的选择

[Ubuntu Mono](http://font.ubuntu.com) 是非常非常不错的字体。前些天我刚从 [Menlo](http://en.wikipedia.org/wiki/Menlo_(typeface)) 切换过来，这绝对不让人后悔。

在我的15寸的MacBook上，Ubuntu Mono的16号字非常适合。1680 × 1050的分辨率对于一个边栏加两个编辑器窗口（自动调整到80个字符宽）刚好合适。

如果你打算认真的挑选一下字体， [slant.co的这篇文章](http://www.slant.co/topics/67/~what-are-the-best-programming-fonts) 写的不错。它包含了大部分流行的编程方面的字体的截图及下载链接。

## 安装插件

正如之前提到的， Sublime 有一个非常丰富的插件系统。而我当前使用的插件如下：

* [Package Control](http://wbond.net/sublime_packages/package_control" \t "/home/cs/Documents\\x/_blank) 在 Sublime 里直接安装附加插件的包管理器。**这是唯一一个你必须手动安装的插件。**这边列出的其他所有插件都可以通过 Package Control 来安装。也可以通过它来更新已安装过的插件。简单得想做是 Sublime packages 的 apt-get 就行了。
* [Color Scheme - Tomorrow Night](https://github.com/theymaybecoders/sublime-tomorrow-theme" \t "/home/cs/Documents\\x/_blank) Color schemes 决定了编辑器界面语法高亮的字体颜色。这是一个非常酷的暗黑系样式。
* [Theme - Soda Dark](http://buymeasoda.github.io/soda-theme/" \t "/home/cs/Documents\\x/_blank) Themes 影响 Sublime 界面元素的颜色和风格。这个非常适合 Tomorrow Night 的配色方案。
* [SideBarEnhancements](https://github.com/titoBouzout/SideBarEnhancements" \t "/home/cs/Documents\\x/_blank) 这个插件提供了侧边栏附加的上下文菜单选项，例如"New file"，"New Floder"等。这些本应当默认就该有的，却没有。
* [All Autocomplete](https://github.com/alienhard/SublimeAllAutocomplete" \t "/home/cs/Documents\\x/_blank) Sublime 默认的自动完成只关注当前文件的单词。这个插件扩展了其自动完成的单词列表到所有打开的文件。
* [SublimeCodeIntel](https://github.com/Kronuz/SublimeCodeIntel" \t "/home/cs/Documents\\x/_blank) 为部分语言增强自动完成功能，包括了 Python 。这个插件同时也可以让你跳转到符号定义的地方，通过按住 alt 并点击符号。非常方便。
* [SublimeREPL](https://github.com/wuub/SublimeREPL" \t "/home/cs/Documents\\x/_blank) 允许你在编辑界面直接运行 Python 解释器。我倾向于在单独的终端窗口用 [bpython](http://bpython-interpreter.org/" \t "/home/cs/Documents\\x/_blank) 来运行，但有时 SublimeREPL 是很有帮助的。
* [GitGutter](https://github.com/jisaacks/GitGutter" \t "/home/cs/Documents\\x/_blank) 在编辑器的凹槽区，依照 Git ，增加小图标来标识一行是否被插入、修改或删除。在 GitGutter 的 readme 中有说明如何更改颜色图标来更新你的配色方案文件。
* [Pylinter](https://github.com/biermeester/Pylinter" \t "/home/cs/Documents\\x/_blank) 这个插件提供了目前我所见到的最好的 [pylint](http://www.pylint.org/" \t "/home/cs/Documents\\x/_blank) 编辑器整合。它自动检查 .py 文件，无论其何时被保存，并且会直接在编辑界面显示 pylint 违规。它还有一个快捷方式来禁用局部的 pylint 检查，通过插入一个 #pylint: 禁用注释。这个插件对于我确实非常有用。

## 配置文件

Sublime Text 的一个优点就是它的所有配置都是简单的基于 JSON 的[配置文件](http://docs.sublimetext.info/en/latest/customization/settings.html)。这使得你可以很容易的将配置转到另一个系统中。我也见过一些人使用 Dropbox 自动同步他们所有电脑上的配置。

**Preferences.sublime-settings** 配置了 Sublimede 的显示和行为.你可以在sublime 中通过 Preferences > Settings — User 打开并编辑此文件。我使用如下配置：

{

// Colors

"color\_scheme": "Packages/Tomorrow Color Schemes/Tomorrow-Night.tmTheme",

"theme": "Soda Dark.sublime-theme",

// Font

"font\_face": "Ubuntu Mono",

"font\_size": 16.0,

"font\_options": ["subpixel\_antialias", "no\_bold"],

"line\_padding\_bottom": 0,

"line\_padding\_top": 0,

// Cursor style - no blinking and slightly wider than default

"caret\_style": "solid",

"wide\_caret": true,

// Editor view look-and-feel

"draw\_white\_space": "all",

"fold\_buttons": false,

"highlight\_line": true,

"auto\_complete": false,

"show\_minimap": false,

// Editor behavior

"scroll\_past\_end": false,

"highlight\_modified\_tabs": true,

"find\_selected\_text": true,

// Word wrapping - follow PEP 8 recommendations

"rulers": [ 72, 79 ],

"word\_wrap": true,

"wrap\_width": 80,

// Whitespace - no tabs, trimming, end files with \n

"tab\_size": 4,

"translate\_tabs\_to\_spaces": true,

"trim\_trailing\_white\_space\_on\_save": true,

"ensure\_newline\_at\_eof\_on\_save": true,

// Sidebar - exclude distracting files and folders

"file\_exclude\_patterns":

[

".DS\_Store",

"\*.pid",

"\*.pyc"

],

"folder\_exclude\_patterns":

[

".git",

"\_\_pycache\_\_",

"env",

"env3"

]

}

**Pylinter.sublime-settings**配置了pylinter 插件。我使用下面的配置让 Pyhton 在保存时自动规范，并对违反规范显示图标。

{

// Configure pylint's behavior

"pylint\_rc": "/Users/daniel/dev/pylintrc",

// Show different icons for errors, warnings, etc.

"use\_icons": true,

// Automatically run Pylinter when saving a Python document

"run\_on\_save": true,

// Don't hide pylint messages when moving the cursor

"message\_stay": true

}

## 按键绑定

Sublime 的按键绑定也是全部可配置的基于JSON的 sublime-keymap 配置文件。我修改了一些默认配置以更好的配合我的 TextMate / IntelliJ 肌肉记忆。你可以完全不修改。如果你想，修改很简单，并可以跨平台使用。我使用如下的绑定：

[

// Rebind "go to file" to cmd+shift+O

{ "keys": ["super+shift+o"], "command": "show\_overlay", "args": {

"overlay": "goto",

"show\_files": true

}},

// Rebind swap line up/down to cmd+shift+up/down

{ "keys": ["super+shift+up"], "command": "swap\_line\_up" },

{ "keys": ["super+shift+down"], "command": "swap\_line\_down" },

// Delete a line with cmd+delete

{ "keys": ["super+backspace"], "command": "run\_macro\_file", "args": {

"file": "Packages/Default/Delete Line.sublime-macro"

}},

// Reindent selection with cmd+alt+L

{ "keys": ["super+alt+l"], "command": "reindent"}

]

## 命令行工具

同 TextMate 的 mate 类似，Sublime Text 包含了一个命令行工具，允许你通过 shell 打开编辑器。工具名为 sublis，默认不可用。要使之生效，在任一 shell 中运行下面：

ln -s /Applications/Sublime\ Text\ 2.app/Contents/SharedSupport/bin/subl /usr/local/bin/subl

要将 Sublime 作为 git 互动命令的默认编辑器使用——举例，撰写提交信息——只需添加下面一行到你的 ~/.profile 文件：

export GIT\_EDITOR="subl --wait --new-window"