**六分钟八法则塑造优秀程序员**

**775个读者** [zhaorui](http://www.yeeyan.com/space/show/zhaorui) @ yeeyan.com 4天前23:12 [双语对照](http://www.yeeyan.com/articles/view/zhaorui/41099/dz)  [原文](http://secretgeek.net/6min_program.asp) 字体大小 [小](JavaScript:doZoom(11)) [中](JavaScript:doZoom(13)) [大](JavaScript:doZoom(16))

简介

如何在六分钟之内学会八种方法，从而帮助你成为更好的程序员？

还记得那个叫做 [Justice Gray](http://graysmatter.codivation.com/) 的人么？他曾经试图在六个月的时间里面成为更好的程序员，这种做法似乎成为了一种众人追捧的[时尚](http://graysmatter.codivation.com/HowIAmBecomingABetterDeveloperPart1OfInfinity.aspx)。

然而很长一段时间过去之后，其中的大部分人并没有如愿以偿的成为好的*[程序员](http://www.yeeyan.com/articles/tag/%E7%A8%8B%E5%BA%8F%E5%91%98" \t "_blank)*。

这里提供给大家一个简单的解决方案，有一些方法可以让你在**六分钟**之内成为更优秀的程序员。

你有六分钟的闲暇时光，对么？

我们一起来吧！

**1. 使用更大的显示字体**

这个办法看上去非常简单，但是的确很有效果。

打开你最喜欢的那个编程工具，将代码编辑器中的字体调大。我把 10pt 的代码显示字体增大到 14pt，这样一来，你的屏幕上一次只能显示出少量的代码。

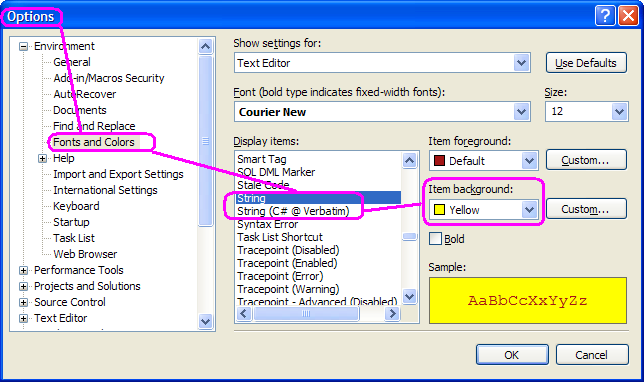
结果：你被迫去写一些更为短小精炼的函数，这无疑是一个好的趋势。

（[Scott Hanselman](http://hanselman.com/blog/) 也推崇这样的做法）

**2. 让代码中的硬编码字符串部分显示变得刺眼**

我是从 [Joe Cooney](http://jcooney.net/) 那里学到这种办法。

在你最熟悉的那个编程工具中将代码编辑器中的字符串设置为更加显眼的格式——比如，红色字符黄色背景。让那些字符串显得非常丑陋、刺眼。这将使你在代码中下意识的少写一些硬编码字符串，并且时刻注意代码中嵌入的字符串。



**3. 找出一个“模糊”的关键字，通过学习熟练掌握**

你是否明白 "yield" 关键字的作用？ 还有什么关键字是你从来没有使用过的么？

编程语言中每一个关键字都有其特殊的用途。学习并掌握这些“神秘”的陌生关键字可以使你的编程能力变得异常强大。

这里有一些 .NET 平台不同语言的关键字列表： [C#](http://msdn.microsoft.com/en-us/library/x53a06bb.aspx), [VB.net](http://msdn.microsoft.com/en-us/library/ksh7h19t.aspx), [F#](http://en.csharp-online.net/FSharp_Functional_Programming%E2%80%94Keywords).

**4. 提高1%的测试代码覆盖率**

不要为了追求 100% 的自动化单元测试代码覆盖率而焦头烂额，但是不妨花费一些时间将你的测试代码覆盖率提高一个百分点。

有可能对你来说，这就是从 0% 到 1% 的过程，然而这已经是一个巨大的进步。

代码中复杂的正则表达式或者重要的业务逻辑必需要有单元测试来确保正确无误。

**5. 阅读一些开源项目的源代码**

当我看到一段完全陌生的代码的时候，通常会有一种奇怪的感觉：似乎是溜进了邻居家的屋子，并且窥视他们家冰箱里面的物品。

我们需要慢慢适应这样的情绪，并且从他人的代码中学习。

也许你可以从这个 [Hanselman's Weekly Source Code](http://www.hanselman.com/blog/CategoryView.aspx?category=Source+Code) 系列开始。

**6. 使用静态代码分析工具来检查自己的代码**

你可以使用 [fxcop](http://msdn.microsoft.com/en-us/library/bb429476%28VS.80%29.aspx)， 或者 [StyleCop](http://code.msdn.microsoft.com/sourceanalysis)、 [clone detective](http://www.codeplex.com/CloneDetectiveVS)、 [ndepend](http://www.ndepend.com/)、 Visual Studio 2008 附带的 [code metrics feature](http://blogs.msdn.com/fxcop/archive/2007/10/03/new-for-visual-studio-2008-code-metrics.aspx), 或者其他自选的静态代码分析工具。

使用这些工具可以发现隐匿在你的代码深处的缺陷。哪怕仅仅是瞥一眼分析结果，也可以让你发现自己的代码还存在很多可以改进的地方。

**7. 找出有“坏味道”的代码进行重构**

其实你自己心理清楚那些“坏味道”在哪里——就是你羞于承认，看上去很可怕，又臭又长，同时对整个应用非常重要的代码。

你不必把那些朽木雕琢成钻石，但是可以把他们变得稍微干净一些——重新命名一些变量，或者将其中的一部分代码独立出来，从简单的操作开始，小心翼翼逐步推进。

**8. 停止阅读，开始编码**

不要仅仅是写那些简单重复的代码，写一个编译器吧！

这篇 ol 写的 MSDN上 的[文章](http://msdn.microsoft.com/en-us/magazine/cc136756.aspx) 是一个非常好的起点。[Joel Pobar](http://callvirt.net/blog/) 将指导你在短时间内写出一个属于自己的语言的编译器。

以上这些就是成为优秀[*程序员*](http://www.yeeyan.com/articles/tag/%E7%A8%8B%E5%BA%8F%E5%91%98)的方法，你明白了么？

有什么方法可以在六分钟之内完成，并且帮助你成为更优秀的[*程序员*](http://www.yeeyan.com/articles/tag/%E7%A8%8B%E5%BA%8F%E5%91%98)么？