链滴

【软文】web 高级开发的成长之路

作者：c3gen

原文链接：https://ld246.com/article/1479194895676

来源网站：链滴

读了这篇文章之后感觉蛮受启发的，在此分享一下，献给和我一样处于困惑的朋友。

正文如下：

本人也是coding很多年，虽然很失败，但也总算有点失败的心得，不过我在中国，大多数程序员都是我一样，在一直走着弯路。如果想成为一个架构师，就必须走正确的路，否则离目标越来越远，正在苦工作的程序员们，你们有没有下面几种感觉?

一、我的工作就是按时完成领导交给我的任务，至于代码写的怎样，知道有改进空间，但没时间改进，关键是领导也不给时间啊。

二、我发现我的水平总是跟不上技术的进步，有太多想学的东西要学，jQuery用的人最近比较多，听说最近MVC比较火，还有LINQ，听说微软又有Silverlight了……

三、我发现虽然我工作几年了，除了不停的coding，Ctrl+C和Ctrl+V更熟练了，但编码水平并有提高，还是一个普通程序员，但有人已经做到架构师了。

四、工作好几年了，想跳槽换个工作，结果面试的考官都问了一些什么数据结构，什么垃圾回收什么设计模式之类的东西，虽然看过，但是平时用不着，看了也忘记了，回答不上来，结果考官说我础太差。。。

有没有，如果没有，接下来就不用看了，你一定是大拿了，或者已经明白其中之道了，呵呵。

如果有，恭喜你，你进入学习误区了，如果想在技术上前进的话，就不能一直的coding，为了完需求而工作，必须在coding的同时，让我们的思维，水平也在不停的提高。

写代码要经历下面几个阶段。

一 、你必须学习面向对象的基础知识，如果连这个都忘了，那你的编程之路注定是在做原始初级重复!

很多程序员都知道类、方法、抽象类、接口等概念，但是为什么要面向对象，好处在哪里，要解什么问题？只是明白概念，就是表达不清楚，然后在实 际工作中也用不上，过了一段时间，面向对象东西又模糊了，结果是大多数程序员用着面向对象的语言做着面向过程的工作，因此要学习面向对象首先应该明白 面向对象的目的是什么?

面向对象的目的是什么?

开发语言在不断发展，从机器语言，到汇编，到高级语言，再到第四代语言;软件开发方法在不断展，从面向过程，面向对象，到面向方面等。虽然这些都在不断发展，但其所追求的目标却一直没变这些目标就是：

1. 降低软件开发的复杂度

2. 提高软件开发的效率

3. 提高软件质量：可维护性，可扩展性，可重用性等。

其中语言的发展，开发方法的发展在1,2两条上面取得了极大的进步，但对于第3条，我们不能光望开发方法本身来解决。

提高软件质量：可维护性，可扩展性，可重用性等，再具体点，就是高内聚、低耦合，面向对象是为了解决第3条的问题。因此要成为一个好的程序员，最绕不开的就是面向对象了。

二、 要想学好面向对象，就必须学习设计模式。

假定我们了解了面向对象的目的，概念了，但是我们coding过程中却发现，我们的面向对象的知似乎一直派不上用场，其实道理很简单，是因为 我们不知道怎么去用，就像游泳一样，我们已经明白游泳的好处，以及游泳的几种姿势，狗刨、仰泳、蛙泳、自由泳，但是我们依然不会游泳。。。。

因此有了这些基本原则是不行的，我们必须有一些更细的原则去指导我们的设计，这就有了更基的面向对象的五大原则，而把这几种原则更详细的应用 到实际中来，解决实际的问题，这就是设计模。因此要学好OO，必须要学习设计模式，学习设计模式，按大师的话说，就是在人类努力解决的许领域的成功方 案都来源于各种模式，教育的一个重要目标就是把知识的模式一代一代传下去。

因此学习设计模式，就像我们在看世界顶级的游泳比赛，我们为之疯狂，为之着迷。

三、学习设计模式

正像我们并不想只是看别人表演，我们要自己学会游泳，这才是我们的目的所在。

当我们看完几篇设计模式后，我们为之精神振奋，在新的coding的时候，我们总是想努力的用上到的设计模式，但是经常在误用模式，折腾半天发现是在脱裤子抓痒。。。

当学完设计模式之后，我们又很困惑，感觉这些模式简直太像了，很多时候我们分不清这些模式间到底有什么区别，而且明白了设计过程中的一个致命 的东西——过度设计，因为设计模式要求我高扩展性，高重用性，但是在需求提出之初，我们都不是神，除了依靠过去的经验来判断外，我们不道哪些地方要扩 展，哪些地方要重用，而且过去的经验就一定是正确的吗？所以我们甚至不敢再轻易设计模式，而是还一直在用面向过程的方法在实现需求。

四、学习重构

精彩的代码是怎么想出来的，比看到精彩的代码更加令人期待。于是我们开始思考，这些大师们非不用工作，需求来了没有领导规定完成时间，只以设 计精彩的代码为标准来开展工作？这样的工作爽了，也不可能，老板不愿意啊。就算这些理想的条件他都有，他就一开始就设计出完美的代码来了也不可能啊， 除非他是神，一开始就预料到未来的所有需求，那既然这些条件都没有，他们如何写出精彩代码？

Joshua Kerievsky在那篇著名的《模式与XP》〔收录于《极限编程研究》一书）中明白地指出：设计前期使用模式常常导致过度工程（over- engineering)。这是一个残酷的现实，单凭对完美的追无法写出实用的代码，而「实用」是软件压倒一切的要素。

在《重构——改善既有的代码的设计》一书中提到，通过重构（refactoring），你可以找出改的平衡点。你会发现所谓设计不再是一切动 作的前提，而是在整个开发过程中逐渐浮现出来。在系统筑过程中，你可以学习如何强化设计；其间带来的互动可以让一个程序在开发过程中持续保有良好的设计。

总结起来就是说，我们在设计前期就使用设计模式，往往导致设计过度，因此应该在整个开发过，整个需求变更过程中不断的重构现在的代码，才能让 程序一直保持良好的设计。由此可见，开发过中需要一直重构，否则无论当初设计多么的好，随着需求的改变，都会变成一堆烂代码，难以维护，以扩展。所谓 重构是这样一个过程：「在不改变代码外在行为的前提下，对代码做出修改，以改进程的内部结构」。重构的目标，就是设计模式，更本质的讲就是使程序的架构 更趋合理，从而提高软件可维护性，可扩展性，可重用性。

《重构——改善既有的代码的设计》一书也是Martin Fowler等大师的作品，软件工程领域的超经典巨著，与另一巨著《设计模式》并称"软工双雄"，不可不读啊。

五、开始通往优秀软件设计师的路上

通过设计模式和重构，我们的所学和我们工作的coding终于结合上了，我们可以在工作中用面向象的思维去考虑问题，并开始学习重构了。这就 像游泳一样，我们看完了各种顶级的游泳比赛，明白

种规则，名人使用的方法和技巧，现在是时候回家去村旁边的小河里练练了。练习也是需要有教练的推荐另 一本经典书叫《重构与模式》，引用他开篇的介绍，本书开创性地深入揭示了重构与模式这两软件开发关键技术之间的联系，说明了通过重构实现模式改善既有的 设计，往往优于在新的设计早期用模式。本书不仅展示了一种应用模式和重构的创新方法，而且有助于读者结合实战深入理解重构和式。

这本书正是我们需要的教练，值得一读。

六、没有终点，只有坚持不懈的专研和努力。

经过了几年的坚持，终于学会了灵活的运用各种模式，我们不需要去刻意的想用什么模式，怎么构。程序的目标，就是可维护性，可扩展性，可重用 性，都已经成了一种编程习惯，一种思维习惯，像我们练习了几年游泳之后，我们不用再刻意的去考虑，如何让自己能在水上漂起来，仰泳和蛙泳的区别..... 而是跳进水里，就自然的游了起来，朝对岸游去。但是要和大师比起来，嘿嘿，我们还有很长路要走，最终也可能成不了大师，但无论能不能成为大师，我们已经 走在了成为大师的正确的路上，们和别的程序员已经开始不一样，因为他们无论再过多少年，他们的水平不会变，只是在重复造轮子唯一比你快的，就是 Ctrl+C和Ctrl+V。

正确的路上，只要坚持，就离目标越来越近，未来就一定会是一个优秀的架构师，和优秀架构师区别，可能只是时间问题。

同样的附上 电子书　重构——改善既有的代码的设计　高清版