# 程序员的谎谬之言还是至理名言？

有朋友（网友never）在酷壳Coolshell.cn的[留言版上问我](https://coolshell.cn/guestbook#comment-40758)，为什么关注了这很多的东西，我想我可以用下文来回答这位网友，因为我和作者的观点几乎一致。这篇文章由 ALAN SKORKIN写的 “[The Greatest Developer Fallacy Or The Wisest Words You’ll Ever Hear?](http://www.skorks.com/2011/02/the-greatest-developer-fallacy-or-the-wisest-words-youll-ever-hear/)” ，我把其全文翻译如下，我很喜欢这篇文章，希望你也喜欢。（翻译的也许不好，欢迎指正）

—————————————————正文分隔线——————————————————

“**I will learn it when I need it –**我会在我需要的时候再学“！我听到这句已经很多年了。这对于一个高速变化的软件行业环境来说，这似乎是一个非常实用的态度。 在某些方面这的确很实用主义，不过在其它的方面，我为这句话感到很不爽。这句话变成了整个软件行业的福音，但却没有让我们的软件行业变得更好。其问题在于，这句话伪装在于其听上去像是一个智慧的有经验的开发者说的，但是人们只是以此为借口而随波逐流。实在是有太多的东西需要我们去了解，我们也的确需要在工作当中来学习这些东西。但是， “在工作中学习”和“根据遇到的问题捡知识”这两者有着巨大的不同。

* 另外，目前整个软件行业越来越需要一堆多面手，也许现在已经是这样了，只是我还没有注意到。当然，我也不喜欢这种情。现在，好像没有人愿意花更多的时间来把某一个东西学好学深学扎实，比如 [计算机科学的基础知识](http://www.skorks.com/2010/04/on-the-value-of-fundamentals-in-software-development/)，或是最新的你正在使用的技术，甚至你在最近几年内每天都在使用其编码的程序语言（参考：[Java is passed by value](http://blog.tmorris.net/java-is-pass-by-value/)）（译注：我在[如何学好C++一文](https://coolshell.cn/articles/4119.html)的回复中已经看到一些这样的人）。何苦呢？你会在你的学习路途中看到这些东西被更新，被废弃，并可能变得小众化。我和很多不同的人讨论过很多次，但是好像没有人意识到这是一个问题。 “哥们，做个实用主义的人吧”。
* 与此同时，我们所有的人都在相互地克隆和模仿（译注：参看[中国的C2C](https://coolshell.cn/articles/3820.html)）。你需要一个Java程序员，我是一个Java程序员，你也是一个Java程序员，我的邻居也是一个Java程序员。我们之间有什么差别？其实，基本没有差别。好吧，我有一些jQuery的经历，太好，所以，你知道怎么来做一个折叠式的菜单？当然，我可以Google一下，然后剽窃别人最好的代码给你 :)（译注：参看“[十条不错的编程观点](https://coolshell.cn/articles/2424.html)”中的”Googling it” is okay）。
* 与此同时，你需要招聘到真正的专业级的人物（比如，你需要写一个很牛的解析程序来以图形化展示一下后台数据），那么你可能需要准备足够的啤酒和三明治，因为你可很有一段时间找不到这样的人。

好了，其实，是有一些方法来区分，比如，我有更好的沟通能力，这就是为什么我可以做得更好。这是相当重要的，但是，如果我们用软能力而不是用技术能力来区分程序员的好坏 –  是不是有那么点反常和变态。我们所有人都可以沟通得很好，但是我们的代码乱成一团 :)。该死的，我本不应该说这事，我也有一点全才的样子。当然，我觉得我自己是一个 [T型人才](http://darrennegraeff.com/the-importance-of-t-shaped-individuals/)（译注：即有深度又有广度），但是如果我们都非常诚实的话，我们会发现更多的人的能力是“横线型”或是“下划线型”的（译注：广度型），其中只有很少数的人才是有能力的。而我们的“T型人才”在这些人中就像一个巨大的钟乳石。你看上去像一个专家，但也许你从没有做过专家，这就是专才在满是全才的世界中的优势。

#### 投资你的未来

我不想以说教的方式来告诉你人们应该怎么样来投资自己未来的职业生涯，因为每个人都知道我们应该怎么做。很多人也许认为他们正在投资，他们努力奋斗，写很多的代码，并也做一些阅读，当然，这样坚持下去，也许[十年也可以成为一个专家](http://norvig.com/21-days.html)，而成为一个咨深专家可能需要20年（我会不断地说这些观点，总有一天我会把这个事说明白 :) ）。但是，如果真是这样的吧，每一个老家伙都会成为各个方面的专家，当然，事实是不会这样的。也许是因为人们不知道如何发展他们的专长（这是事实），但我私下里却怀疑，大家缺少的是热情而不是知识。我所说的这些东西难道不都是这样的原因吗？

我完全跑题了。“在专业上投资未来”只是这些流行语中的一个，而重点是那句话 – “**I will learn it when I need it –**我会在我需要的时候再学”。对我老爸来说这样做挺好，迄今为止对我也有效。但是让我们换一个角度看看，比如金融方面，如果我们说：“我会在我需要钱的时候再去进行投资”。在这句话上面那个实用主义的光环就不那么漂亮了。

#### 你不知道你不知道的

我们都有过这样的时候，当我们痛苦地在解决一个问题的时候，有人突然告诉你一个算法或是一个技术，其把这个问题变得又快又简单。是的，有人告诉你一个容易的方法是一件很幸运的事，否则你可能需要花上数天或数周才能找到解决方法，并且事情可能会变得一团糟。你不会因此而被批评，因为你不知道你所不知道的东西。对此，“**I will learn it when I need it –**我会在我需要的时候再学”在这个时候就走不通了。因为，你不可能学习那些你以为不存在的东西。Google做了很多的工作来减轻和缓解这样的问题，但并不完美。在一个陌生的环境下面对着一堆陌生的问题，会让你非常非常地痛苦，除非你知道你所面对的是什么（例如：如果你对搜索和约束传播（constraint propagation）有点了解的话，那你就可能 [容易地解决数独问题](http://norvig.com/sudoku.html)，否则的话， [这的确是](http://xprogramming.com/xpmag/OkSudoku) [非常](http://xprogramming.com/xpmag/SudokuMusings) [难的](http://xprogramming.com/xpmag/Sudoku4)）。你无法学习一种不知道或是你不知道用在哪里的算法。你也不可能去用一个你以为不存在的技术去解决一个问题。同样，你身边也不可以一直都有一个高人随时在给你指引正确的方向。我敢打赌，这个世界上有几十亿行代码可以被 几百万行更快，更清楚，更简单的代码所取代，因为无论是谁来写，他都不可能知道他所不知道的东西。

我想在这里成为这个观点的反方，如果我们知道我们有哪些东西我们不知道，那么我们就知道我们需要去关注哪些东西。粗浅地尽可能的大范围的知道这些东西，那么，我们就可以在遇到问题的时候明白我们应该去更深地学什么样的知识。但是，这样来做在实际上并不有效，因为，这些浮云般的知识不会让你记下来，我们的大脑不是这样工作的。如果我们不去强化或是深度挖掘去消化这些概念， 我们的大脑会很快地 [把这些信息标为不重要并换页出去](http://www.skorks.com/2009/09/become-a-better-developer-by-indexing-your-brain/)，这么做无非就是在浪费时间（你回头想一想你那些“填鸭式”的死记硬背的知识，你今天还记得吗？）然而，对于那些你集中精力深度研究过的东西——并伴随着你的兴趣的东西——你会收获到扎实的知识（那是你大脑里实际存下的不会忘的东西）。我的爷爷是一个核物理学家，数十年的在工作中获得这个领域中更深的知识让他今天成为了一个专家，同时也让他成为了一个优秀的数学家，不错的化学家，非常好的地理学家，还是一个合格的生物学家，等等。只需要一些 [观察性的证据](http://en.wikipedia.org/wiki/Empirical) 你就知道广度的知识是深度研究的副产品。

#### 你学得足够快吗？

有些东西你需要花很长的时间才能学会。我对自己有信心不需跳跃性的就能把一个我从没有见过的ORM框架学会，因为我以前用过相似的东西，它们在概念上是相同的。但是，如果你需要做一个把演讲转成文本的东西呢，这并不简单，因为你没有足够多的背景知识。你可以希望通过Google给你提供一些东西让你Copy/Paste，但这是一个很不好的做法，只有大学里的做研究的研究员干这种烂事。如果是要创建一个网站呢，我们都知道怎么去创建一个网站，但是有多少人会知道如何架构一个每天有一千万用户访问的网站？那么我们需要去学的就是怎么做扩容扩展，我相信你的用户需要等你一到两个月才能把速度提上去:)。是的，我太笨了，所有我需要做的就是去招一个专家，然后……嗯……哦等一下，我们的啤酒和三明治都不够了。

#### 为什么我应该关心

和高手在一起工作真是超爽无比。你也许以前经历过，他们每说的一件事总是新鲜的，总是有意思的，你能从他们每一行的代码中学到很多小技巧，你几乎可以感觉到你的大脑在不断膨胀:)。你想从高手学习，所以，如果你身边没有高手，那真是太糟糕了。因为每一个人只会去学那些“需要”被学的东西，所以没有人能教给你任何有意思的东西。然而，这些高手也总是想和高手一起工作， 所以，你需要做的事就是确定能让高手想和你一起工作？。按需所学也许是一个不错的技能，但其不应该成为程序员的价值观。是的，这是一个巨大的行业你不可能学习所有的东西，所以，你需要有所选择地把其学精，只要你有足够的好奇心去跟从你的兴趣，你会发现最终你会真正掌握很多很多其它的东西。如果你能把你的工作做好，那么其它的超级牛人都会想要和你一起工作，因为他们可以从你这学到东西，而你又可以从他们那里学到东西。这样一来，所有的人都会是成功者。

Image by [SamueleGhilardi](http://www.flickr.com/photos/samueleghilardi/2971657900/) and [SpecialKRB](http://www.flickr.com/photos/specialkrb/3250756763/)

—————————————————正文结束分割线——————————————————

我在这里想说几个我的观点：

1. 我特别同意作者的，如果你把一个技术搞精搞深，你的知识面自然会很广的。
2. 面对于各种比较深的东西（比如C++的奇技淫巧），作为一个实用主义者可能很不屑，但是你也会为此而失去开阔眼界的机会。
3. 为明天做一些技术储备，因为你不知道你所缺的东西。多多阅读，多多交流，最好能把自己的心得写下来强化自己的认识和记忆。
4. 不要只寄望于在工作中学习，工作没有覆盖的地方你就不学了。真正的高手在工作之余都会花很多时间去自己研究点东西的。
5. 永远和高手一起工作。如果你面试的公司的面试太简单了，那就不要去，因为简单的面试通常意味着平庸。去那样的公司工作只会让你的学习速度变慢，甚至倒退。
6. 很多东西在概念上是相通的，在哲学层次上是相通的，这是你需要去追求的学习知识的境界。
7. 最后echo一下作者的话——“很多时候，你缺少的不是知识而是热情”！

谢谢大家又花了一点看我的唠叨。呵呵。