Steve Yegge：Google面试秘籍

投递人 itwriter 发布于 2013-05-16 17:15 评论(3) 有2322人阅读 原文链接 [收藏] « »

英文原文：steve yegge，编译：@老码农的自留地

我憋了很长时间想写点关于去 Google 面试的秘籍。不过我总是推迟，因为写出来的东西会让你抓狂。很可能是这样。如果按统计规律来定义“你”的话，这文章很可能让你不爽。

为啥呢？因为啊……好吧，对此我写首小诗回答：

哎妈呀，俺咋听不懂涅

这个史迪威讲的都啥啊

要是俺老板也脚底他对

俺的工作就得玩儿完啦

哎妈呀，哎妈呀.....

你们感受一下。

（本文作者：Steve Yegge，业界大牛程序员，Google 员工）

当我还在别的公司，刚开始写点有关面试的东西的时候，我根本没有意识到会有上面这种典型的对面经的反应。不过很快我还是发现了。

看吧，大概的情况会是这样的：

我：blah blah blah， 我喜欢在面试里问X这个问题，blah blah blah…

你：X这个问题？哥们，我自打上大学就没听说过X！我的工作中也永远不需要用到它！他在面试里居然问这个？不过这说明那里有人觉得知道它很重要，而且，而且…我却不懂这个！如果他们发现了我在这方面的无知，不仅我会被现在的公司无情地扫地出门，而且那些喜欢问X问题的面试官也都会拒了我！如果人们都听信史迪威的话，那所有的面试官都不会招我了！我将无家可归，穷困潦倒！仅仅是因为我没有懂一些我之前永远用不上的东西！这太可怕了！我应该贬低X这个东西，除此之外我并不想先找本书仔细研究明白它再去否定它。显然我必须到处宣传史迪威是如何愚蠢，这样就没人会听他的了！

我：所以总体来说，blah blah… 嗯？你刚才说什么“扫地出门”？还有“穷困潦倒”？你在说什么啊?

你： 嗷！左一刀，右一刀，我刀刀不离你的后脑勺！

我： 好吧。我不会再谈面试的事了。

24小时阅读排行

高清直播世界杯 优酷是如何做到的
万科来到城中村 富士康员工慌了
华谊兄弟的至暗时刻：“兄弟”出走 又掀质押风波
为什么大多数的A/B测试都不靠谱?
库克：我不后悔公布自己是同性恋
浑水猎杀中国市值最高教育上市公司，好未来未来在哪
阿里为什么不会放过拼多多?
更多...

最新新闻

618期间京东带动线下零售全面爆发 步步高日均增长…
800亿网贷平台爆雷！自称央企、高额返现、还玩虚…
阿里为什么不会放过拼多多?
连PS都自叹不如！Facebook的这项 AI 眼部修复技…
618购物节前夕，我发现了一个薅羊毛群
GitLab发布Web IDE
微信被指封杀Xposed框架用户
更多...

相关新闻

谷歌地图向部分用户显现快速访问按钮 一键导航回家
法院裁定谷歌翻译具有局限性“欺骗”男子同意警方…
谷歌推出VR180转换/发行工具VR180 Creator 仅提…
喜迎GIF 31周年：谷歌在GitHub上发布开源的CLI终…
谷歌拟在非洲大陆设立首个AI研究中心
Google低代码工具App Maker正式对外开放，不写…

到底X是啥并不重要。它可能是任意一个概念。我可能会这么说：“在面试中向应聘者发问真的感觉好爽哦”，可是应聘者们还是会被吓坏了，因为他们对面试这档子事或者对于自己的名字缺乏安全感。但愿是前者吧。

可是，然后呢，随着时间流逝，面试者们来来去去，最后我们总是说：“天啊，我们真的希望刚才那位显然很聪明的应聘者对于面试准备得更好一点。我们有什么办法整点小秘籍来帮助未来的应聘者们呢？”

然后也没谁真的去做点什么，因为我们都害怕被某些不懂X概念的人给凶狠地砍上几刀。

我也考虑过只给出一套秘籍，里边就用类似于X这样的变量名而不是真正的主题，不过还是觉得在这样产生的真空状态下，每个人恐怕都会抓狂。不然的话，那个方法看起来还是挺不错的，只要我用一个假名发布它就好了。

最后，应聘者们真的需要一些秘籍，不管看秘籍的时候会有多么揪心。所以，与其绕弯子扯闲淡，我还不如实实在在地告诉你们一些具体的重要的X概念，还有不少有关面试准备的信息。

附加免责声明

本博客和 Google 无关。Google 不知道我在发布这些秘籍。这只是你我之间的事情，好不？别告诉他们我帮你支招了。你就直接去搞定面试，这样咱俩都是正人君子。

这里我只谈一般性的软件工程师职位及相应的面试环节。

这些秘籍其实是通用的，没有什么特别针对 Google 而不适用于其他软件公司的东西。其实我早就可以写点针对 20 年前我的第一份软件开发工作的秘籍。也就是说，这些秘籍无关时间，至少在我们的职业生涯期间是无所谓的。

显然这些秘籍本身也不会让你得到工作。我的愿望是你遵循了这些秘籍能在面试中表现出你的最佳水平。

噢，呃，那么为啥要提到 Google 呢?

啊哈，你问我为啥提到 Google？好吧，那咱们就马上开始对话吧，好不？

你：我应该去 Google 工作吗？就因为他们说该去，还是有啥别的好处？我在那里会不会平安快乐？我是否应该马上就申请?

我：是的。

你：该哪个问题了……等等，你说“是的”是什么意思？我都还没告诉你我是谁呢！

我：哥们，答案就是“是的”。（你也可能是个女的，不过我还是要叫你哥们。）

你：可是……可是……我的惰性大到接近瘫痪的状态，觉得在现在的公司呆着挺舒服的，或者说至少我已经对它不舒服的地方有了一定忍耐力。我认识这里的很多人，而在 Google 我谁也不认识！我可能不得不去学习 Google 的 build 系统，还有技术什么的玩意儿！我在那里没有任何的信任和声誉，我可能必须要从头开始！我等得太久了，而那里根本没有上升空间！我害怕……

我：哥们，答案已经是“是的”了，对不？这是一个常量了。其他去 Google 的每个人当时也是处于和你一样的境地，除了一小撮长着让甘道夫都自惭形秽的大胡子的名人们。可是他们只是极少数而已。每个申请了的人都有和你一样的不去申请的理由。而且这里的每个人都会说：“天啊，我真的很高兴来了这里工作！”所以只管申请就是了，不过要先准备好。

你: 可是，要是我被误判了咋办？我可能既聪明又称职，可是因为某种说不清道不明的原因，我可能在面试里表现很差然后被拒！那对我幼小的心灵是多么巨大的打击啊！与其接受失败的几率，我还不如完全放弃这个机会呢！

我：没错，这种估计至少是部分正确的。我第一次面试就基本没搞定，不过我当时像条流浪狗一样苦苦哀求，直到他们给了我第二轮面试的机会。我抓住他们心理脆弱的时机搞定了他们。在第二轮面试里，我预先准备好了，表现就大有改观。

重点是，Google 有业内著名的假阴性率，这意味着我们有时候会拒掉合格的应聘者，因为这样感觉比有时招进来不合格的人更好一些。这问题实际上在整个行业里都普遍存在，只是比例在不同公司里有差别而已。在 Google，假阴性率是相当高的。我不太具体知道有多高，但是我确实知道很多聪明且合格的应聘者没通过我们的面试。这是无奈之举。

不过真正重要的道理是：如果你没拿到 offer，你还是可能胜任来这里工作的。所以，不要为此给你幼小的心灵造成巨大的打击。

我认识的任何人都知道，假阴性是完全随机出现的，并且和你的技能或资质无关。假阴性结果可能来自一系列因素，包括但不限于：

1. 你正好今天不爽
2. 一个或更多面试官正好今天不爽
3. 在你和一个或几个面试官之间的沟通有一些微妙的问题
4. 你不走运，碰到了面试敌对圈

啊不，不要碰到面试敌对圈！

是的，恐怕你需要担心这个。

这是什么，你问我？好吧，回想我还在亚马逊工作的时候，我们当时进行了（毫无疑问，他们现在也还在进行）大量的对于这个问题的研讨。我们最终的结论是每个亚马逊的员工E都会有至少一个”面试敌对圈“：一批会在面试中拒掉E的其他的员工S。理解其根本原因对应聘者是很重要的，所以我会告诉你一些我在这些年的发现。

首先，你不能告诉面试官什么重要什么不重要。在任何公司都不行。除非他们明确地征求你的意见。在一位工程师从大学毕业后，只有一年左右的非常窄的时间窗口可以对其进行有关面试的教诲，在那之后时间窗口就会关闭，然后他们相信他们已经是一个”优秀的面试官“了，从此不再改变他们的问题，提问的风格，面试的风格，或者他们提供反馈意见的风格，永远不会再改变了。

这是个问题。不过我已经碰壁多次，不再尝试（去改变他们的想法）了。

第二个问题：每个“有经验的”面试官都有一套自己钟爱的主题，还可能还有一些具体的问题他们认为可以用来准确判定应聘者的能力。任何两个面试官的题库可能会大相径庭，甚至完全没有交集。

一个随处可见的典型例子是这样的：面试官A总是考 C++ 细节，文件系统，网络协议和离散数学。面试官B总是考 Java 细节，设计模式，单元测试，Web 框架和软件项目管理。对于任意一个给定的应聘者，如果面试官A和B都在面试圈里，他们很可能会给出非常不同的评价。其实如果碰巧的话，当年A和B甚至很可能都会在面试中互相拒掉对方，但正好他们应聘的时候碰到的都是面试官C，他给俩人出的题是有关数据结构，Unix 实用工具以及进程和线程，对这些A和B正好都在行。

那几乎就是你从一个高科技公司拿到 offer 时所发生的一切：瞎猫碰到了死耗子。由于这种先天有内在缺陷的面试过程，很有可能某个在面试圈里的人会对你印象不佳，即使你就是阿兰·图灵。实际上，如果你是阿兰·图灵就更为不妙，因为这意味着……显然，你不会C++。

这里的底线是，如果你要去参加任何软件公司的面试，你必须做好预案以应对你走背字的情况，届时你会碰到一个或多个你的面试敌对圈人士进入你的面试官行列。如果这种情况出现了，你会努力再努力，然后被告知你这次不适合申请的岗位，然后你会感到不爽。只要你不是觉得极其不爽，事情就没啥大不了的。你应该为发生了这样的事情后你觉得不爽而高兴，因为啊，这说明你是正常人类。

之后，你应该等上 6 到 12 个月然后再申请。那基本上是我（或者我认识的任何人）所能想出的对付假阴性问题的最佳解决方案了。让我们忘记过去，从头再来。这里有很多人都是在第二次或第三次尝试时被录用的，他们现在都干得很不错。

你也能做到的。

对了，对于潜在的被拒风险我感觉好多了。

不错！那就让我们开始了解这些秘籍吧。

如果你刚才读的够仔细的话，你会意识到我就是面试官D。这说明我个人偏爱的问题和主题不过是我个人的，和其他人的偏好相比，谈不上好也谈不上差。所以我没法告诉你这些问题和主题是什么，无论我是多么愿意告诉你。因为这样做会得罪喜欢其他问题的面试官A，B，C…一直到面试官X。

恰恰相反，我想帮你准备好一些常规话题，这些话题我相信在类似于 Google 的公司里绝大部分的面试官都会用到。大概说来，这种公司会自己开发很多软件，并且进行大量的分布式计算。也有其他的一些技术公司，反差最大的哪一类是把所有事情都外包给顾问，并尝试使用尽可能多的第三方软件的那些公司。我的秘籍仅仅对类似于 Google 的公司有效。

所以你可能也可以把它当做就是 Google，对不？

首先，我们来讨论非技术性的准备。

热身

没人会不热身就走上拳击台。秘籍：你可以带着拳击手套去参加面试。不，等等，不好意思，我的意思是说要提前热身。

如何热身呢？它基本上可以分成短期热身和长期热身，两种你都要进行。

长期热身的意思是：在面试前花一到两星期进行学习和练习。你需要让你的思维进入在白板上解答问题的一般“模式”。如果你能在白板上搞定问题，所有其他的媒介（笔记本电脑，共享网络文件，以及其他）就是小菜一碟了。所以要以白板为主攻方向。

短期热身的意思是：在面试前一天晚上充分休息好，然后在面试当天的上午做大量快节奏的热身（译者注：就是做题）。

我所知道的两个最好的长期热身方法是：

1）找一本数据结构和算法的书复习。为什么？因为这最有可能帮助你提升识别问题的能力。当你不需要更多解释就理解了面试官问题是哪一大类的时候，很多面试官都会觉得满意。例如，如果他们问你一个有关用不同颜色填充美国地图上各个州的时候，如果你认识到这是一个图论里的涂色问题，你会得到加分，即使你已经不记得涂色算法是怎么弄的。

如果你记得它是怎么弄的，那你可能会很快搞定答案。所以从面试准备角度来说，你最好的准备工作就是练习识别问题的技巧，弄清每类问题最适合用哪些算法和数据结构来解决。

对于这类面试准备我绝对钟爱的是 Steven Skiena 写的那本《算法设计手册》。对于我理解图问题是多么惊人的普遍存在（和重要）方面，它比其他任何书都更有帮助，所以我觉得每个程序员的工具箱里都应该有一本。这本书也介绍了基本的数据结构和排序算法，也算是物超所值了。不过真正的金矿在该书的后半部分，里边是单页的百科全书，覆盖了无数有用的问题以及解答它们的各种方法，简明扼要。几乎每个单页都有一个简单的图片，使之易于记忆。这对于学习如何识别数以百计的问题类型是一个很棒的方法。

我认识的其他面试官推荐《算法入门》。这是真正的经典和无价之宝，不过复习它你可能需要多于两周的时间。如果你想胸有成竹地去面试，也可以考虑推迟你申请的时间，直到你掌握了该书的精髓为止。

2）找个朋友模拟面试。该朋友要随机地问你一些面试题，你则在白板上写出解答。不管你觉得多么疲劳或懒散，坚持做完所有面试题。只要你能承受就尽可能多地模拟。

在我第一次去 Google 面试前我没做过这两种准备，结果我非常震惊地发现我在白板编程时表现得巨差无比，因为我已经 7 年没参加过面试了。这很难。而且，我曾经知道或者至少听说过的一堆算法和数据结构都全忘光了。

通过花一周时间进行上面这些练习，我就对第二轮 Google 面试准备得很充分了。然后我在第二次面试里的表现就大有改观。是这些练习改变了一切。

对于短期准备，你能做的无非是让自己尽可能保持状态。不要没热身就开始。先解答一些问题，浏览你的复习材料。喝点咖啡，它有助于让你思维更敏捷。信不信由你，反正我是信了！确保在你开始面试前至少花了一个小时进行练习。把面试当做一场体育比赛或者音乐会，或者一次考试，总之，如果你先热身了，那么你就能表现出最佳状态。

心理准备

好了！你现在是一位编程高手，有一长串的成就。现在是忘记所有这一切而只关注通过面试的时候了。

你必须以谦虚、开明和专注的形象出现。

如果你表现出傲慢，面试官会琢磨他们是否愿意和你一起工作。最常见的傲慢表现就是质疑面试官提出问题的合理性，这无疑会让他们不爽，正如我之前指出的。记得我说过你不能教面试官如何面试吗？没错，如果你是应聘者，这一点更是真理了。

所以，别这么问：“天啊，算法真的那么重要吗？你在现实生活中有过需要那么干活的情况么？我永远不需要去干那样的活。”你这么问只会被拒，所以别问那样的问题。把每个问题都当做合理的来看待，即使你因为不知道如何解答而备感沮丧。

如果你钻到牛角尖里出不来了，可以寻求帮助或提示。有些面试官会因此给你减分，不过，面对那些可能让你陷入半小时可怕的沉默的问题，这么做偶尔能帮你突破一些困难，让你得到优秀的成绩。

当你在思考的时候嘴里别念念有词。

不要试图改变主题去回答一个不同的问题。不要试图通过讲战争故事转移面试官的思路，让他没法问你问题。不要试图恐吓你的面试官。你应该专注在他们问你的每个问题，尽全力完整地解答它们。

有些面试官不会硬性要求你写代码，不过他们往往期望你在解答过程中某个时间会在白板上写代码。他们会提示你，但可能不会直截了当地说：“现在我要求你在白板上写一些代码。”如果你不确定，你应该问他们是否希望看代码。

不同的面试官对于代码的要求也相去甚远。我自己是不太在乎语法的（除非你写的东西明显不可能在任何编程语言里实现，这时候我会插话，确认你实际上并不是马戏团变戏法的小丑，而这部分代码的确是错的）。可是有些面试官对于语法是很挑剔的，有些甚至会因为你少写了个；或者｝就默默地把你否了，而且还不告诉你。我认为这些面试官是.....好吧，这是个技术性词汇，和“魂淡”发音差不多，可他们觉得自己是绝顶的技术评估高手，而且咱也没办法和他们说理呀。

所以还是要问。问他们是否重视语法，然后，如果他们重视，尽力把语法写对。从不同角度和距离仔细检查你的代码。假装这是别人的代码，而你被安排来发现其中的 bug。当你站在距离白板 2 英尺的地方，而面试官盯着你的肩胛骨的时候，你会惊奇地发现自己居然会在某些地方出错。

问一些澄清的问题是可以的（而且是相当受鼓励的），偶尔和面试官确认一下你解答的思路是正确的。如果你听完问题就跳起来开始写代码，即便你写对了，有些面试官还是会给差评。他们会说你没有先考虑清楚，而且你是那种“咱用不着做设计”类型的牛仔程序员。所以即使你觉得你知道问题的解答方法了，在一头扎进去之前最好先问一些问题，并讨论一下你打算采用的方法。

在节奏方面，不要在实际解答问题之前消耗太多时间，不然一些面试官会给你耽误时间的罚分。尽可能快地解答和写代码，因为面试官通常都希望在面试中多考一个问题，如果你解答第一个问题太慢，他们就没有时间再考了。他们会把你否掉，因为他们没法完整地了解你的技能。在面试中很少看到争议带来的好处。

最后一个非技术秘籍：随身带着你自己的可擦除白板马克笔。在办公文具店里有那种细笔芯的马克笔卖，而大部分公司（包括 Google）基本都买的那种粗笔芯的。细笔芯马克笔可以把你的白板从 480i 标准清晰度显像管变成 58 英寸 1080P 高清等离子体液晶屏。在面试中，你需要所有能获得的帮助，而充分的白板空间是一个真正的福音。

你也需要练习白板空间管理技术。例如，不要从右边开始一直把代码写道右下方的角落，最后写成了蝇头小字。你的面试官不会对此留下好的印象。有意思的是，虽然我对应聘者这么干很生气，我自己面试的时候也这么干过。当心这个问题。

哦，还有，别站在那里挥舞你的马克笔，它会挥发最后变干的。我这是在提醒你：尽可能减少面试中让人分心的事情，而那也是不可思议的常见的例子。

好了，非技术秘籍告一段落。现在来说说X，给X赋点值！别砍我啊！

技术性准备秘籍

最好的秘籍是：去读个计算机科学的学位。你有的计算机科学知识越多越好。虽然你并不一定必须有计算机专业的学位，但它对你应聘有利。你也不一定非要有个研究生学历，但它也对你应聘有利。

不过，这是 2 到 8 年之后的事了，你很可能在考虑在更早一点的时间申请 Google 的工作，所以下面是一些短期的秘籍。

算法复杂度：你需要知道O()的概念。必须滴。如果你还在苦苦挣扎于理解基本的算法复杂度分析概念，那你基本上肯定无缘这份工作了。这个概念应该是计算理论教材最开头一章的内容，所以赶紧去学吧。你能行的。

排序：知道如何排序。不要用冒泡排序。你必须知道至少一个n*log (n)复杂度排序算法的细节，最好知道两个（比如，快速排序和合并排序）。合并排序在快速排序不适用的情况下会非常有用，所以要好好看一下。

看在上帝的份上，不要在面试中试图对一个链表进行排序。

散列表：按理说散列表是人类所知的最重要的数据结构了。你绝对必须了解它们的原理。同样，这也会占数据结构书里的一章，所以要去好好学一下。你必须能在一次面试的时间里，用你最喜欢的语言，做到只用数组去实现一个散列表，

树：你必须了解树结构。我要说的是：这是基本的东西，其实在这里提起它显得挺没档次的，可是你们中的一些人啊，竟然不知道基本的树的构造，遍历和其他操作算法。你最起码最起码必须熟悉二叉树，N叉树，还有字典树。树结构很可能是你们长期热身练习最好的练习题来源。

你必须熟悉至少一种类型的平衡二叉树，无论是红/黑树，伸展树还是 AVL 树。你必须准确地了解它是如何实现的。

你必须了解树的遍历算法：广度优先和深度优先，并知道中序、后序和前序的差别。

你每天可能不会用到很多树结构，但是如果真的是这样，那只是因为你在回避树结构问题。一旦你学会了它的原理，你就无须再那么做了。学吧！

图

图结构是非常非常重要的。比你想象的还要重要。即使你已经觉得它很重要，可能它实际上比你想的还要更重要。

在内存中表达图有三种基本形式（对象和指针，矩阵，以及邻接表），你必须了解每一种形式并知道各自的优缺点。

你必须了解基本的图遍历算法：广度优先搜索和深度优先搜索。你必须了解它们的计算复杂度，它们的权重，以及在真实代码中如何实现。

你必须尝试学习更潮的算法，例如 Dijkstra 和A*，如果你有机会的话，它们在任何地方用都很好，从游戏编程到分布式计算，随你说。你必须了解它们。

任何时候别人给你一个问题，用图结构去思考。图结构是最基本和最灵活的表达任何关系类型的方式，所以任何有意思的设计问题都会有 5 成可能要用到图结构。在转向其他解决办法之前，先完全确定你想不出用图结构解答它的办法。本秘籍非常重要。

其他数据结构

你应该学习尽可能多的数据结构和算法，直到你的脑袋装满为止。你应该特别了解 NP 完备性问题中最著名的几类，例如旅行销售员问题和背包问题，能在面试官隐蔽地问到你这类问题时把它识别出来。

你必须搞明白 NP 完备性是什么含义。

总体来说，努力学习数据结构，尝试尽可能多记，这总是没错的。

数学

有些面试官会问基本的离散数学问题。在 Google 这类问题比在我呆过的其他地方更流行，我觉得这是件好事，虽然我在离散数学方面也不是特别牛。我们身边都是计数问题，概率问题，和其他离散数学的问题，而我们身边那些不懂数学的人在轻率地编一些他们自己都不知道是什么的程序。

如果面试官考你数学题，别抓狂。尽最大努力去做。如果你在面试之前花点时间复习组合数学和概率论的话，你的最大努力会更得力。你应该熟悉n选k问题以及同类的问题，越多越好。

我知道了，我知道了，你没多少时间。不过这里的秘籍能帮你实现从“我们不太确定”到“我们招她吧”的飞跃。而且情况也不是都那么糟糕，离散数学没用到多少你学过又忘了的中学数学。它其实是起始于小学数学然后扩充出来的，所以你很可能通过几天的努力学习重新捡起面试需要的那些知识。

很遗憾，对于离散数学教材我没有什么好的推荐，如果你有，请在评论里提出。谢谢。

操作系统

这是我补充的一点，它让你了解进程、线程和并发问题。很多面试官会考那些概念，而且这是很基本的，所以你应该了解它。要了解锁、互斥、信号和管程，以及它们的工作原理。要了解死锁和活锁以及如何避免它们。要了解进程和线程分别需要什么资源，context 切换如何进行，它们在操作系统以及底层硬件是如何初始化的。了解一点计划任务。世界正向多核系统迈进，如果你不了解现代并发架构的话，你很快就会变成落后于时代的恐龙。

对于该主题我自己读过的最好最实用的书是 Doug Lea 写的《Java 并发编程》。它里边每一页都令我拍案叫绝。显然还有很多其他关于并发的书。我一般会躲开太学术性的，专心于实用的东西，因为这些概念最可能被在面试中考到。

编程

你必须精通一门编程语言，最好是 C++ 或者 Java。 C#也行，因为它和 Java 非常相似。至少在一些面试中，你需要写一些代码。你应该对你擅长的编程语言了解很多细节。

其他事务

由于我在之前阐明的一些规则，你还是有可能碰到面试官A，你按我的秘籍复习的东西没有一样用得上（除了热身以外）。如果是这样，就尽最大努力去做。即使碰到最坏的结果，你还总是可以过 6 到 12 个月再杀回去，对不？看上去这时间挺长，不过我向你保证它会一闪而过。

实际上，我谈及的内容大部分是警示性的：如果你不了解它那就大事不妙了。离散数学也许是可选的，不过你如果不了解最基本的东西也有点危险。我提到的其他所有东西你都应该了解，然后你至少要准备到基准面试的水平。在实际面试中出的题可能会比基准面试难很多，也可能会容易一些，基本上取决于面试官的情况。

这就看你有多幸运了。你感觉要走运了？那就去试试呗！

来自: blog.jobbole.com

标签： Google 程序员

« 上一篇：很有爱：第一代iMac变猫窝(2013-05-16 17:09)
» 下一篇：Firefox代码的可维护性(2013-05-16 17:16)

已经有 3 位园友对此新闻发表了看法。

第1楼 l' koni 发表于 2013-05-16 19:00

> 应该是老外写的，很随意。。。

支持(0) 反对(0) 回复 引用

第2楼 FX夜归人 发表于 2013-05-17 09:28

> @l' koni
> 引用
> 应该是老外写的，很随意。。。

聪明绝顶了啦，这都被你看出来了

支持(0) 反对(0) 回复 引用

第3楼 SamHong 发表于 2013-05-17 09:40

神翻译。。。

支持(0) 反对(0) 回复 引用

刷新评论

注册用户登录才能发表评论，登录或注册。