**《黑客与画家》摘抄**

前言

软件带来财富，仅仅代表了大趋势的一面而已。这种大趋势就是本书的主题。我们的时代是计算机时代。以前，人们曾经认定这个时代应该是太空时代或者原子时代。但是事实证明，它们只是公关公司发明的概念。计算机对人类生活的影响远远超过了太空航行或者原子技术的影响。

计算机程序只是文本而已。你选择什么语言，决定了你能说什么话。编程语言就是程序员的思维方式。

为什么黑客那么在乎言论自由？我认为，部分原因在于，革新对于软件行业实在是太重要了，而革新和异端实际上是同一件事。优秀的黑客养成了一种质疑一切的习惯。这是肯定的，因为如果你不得不同一台机器打交道，而这台机器全部由文字组成，像机械式手表一样复杂，并且规模大出1000倍，那么你也会养成这种习惯的。

1 为什么书呆子不受欢迎（他们的心思在别的地方）

所以，如果智力本身与“受欢迎”无关，为什么聪明的小孩一直不受同龄人的欢迎呢？我认为，答案就是他们真的不想让自己受欢迎。

虽然“书呆子”饱尝不受欢迎之苦，但是为了解除痛苦而让他们放弃“聪明”，我想大多数人是不会愿意的。对他们来说，平庸的智力是不可忍受的。我认为，这就是问题的根源。“书呆子”的目标具有两重性。他们毫无疑问想让自己受欢迎，但是他们更愿意让自己聪明。

同伴的意见成为他们判别事物的标准，这不仅体现在衣着上，还体现在他们做的几乎每一件事情上，就连走路的姿势也不例外。所以，他们为了把所有事情“做对”，所付出的任何努力，不管是有意还是无意，实际上都等同于努力在使自己变得更受欢迎。

书呆子不受欢迎的真正原因，是他们脑子里想着别的事情。他们的注意力都放在读书或者观察世界上面，而不是放在穿衣打扮、开晚会上面。

在一个人产生良知之前，折磨就是一种娱乐。

难怪聪明的小孩读中学时往往是不快乐的。他们有其他的兴趣，没有多余的精力用来使自己更受欢迎。你在其他地方有所得，就会在这个地方有所失。不受欢迎使得书呆子成为全校攻击的目标。令人惊奇的是，这种噩梦般的情景并非出自任何有预谋的恶意，而仅仅因为这个特殊的环境。

没错，成年人不知道孩子们内部发生的事。认识到这一点很重要。在抽象意义上，成年人知道孩子的行为有时是极端残酷的，这正如我们在抽象意义上知道贫穷国家的人民生活极端艰难。但是，像所有人一样，成年人不喜欢揪住不放这种令人不快的事实。你不去埋头探寻，就不会发现具体的证据，就会永远以为这件事是抽象的。

我认为，真实世界的关键并非在于它是由成年人组成的，而在于它的庞大规模使得你做的每件事都能产生真正意义上的效果。

真实世界的特点是，它极其庞大。如果总体足够大，即使是人数最少的少数派，只要聚集在一起，也能产生可观的力量。

如今的成年人根本不接受青少年。一般来说，他们都是在办公室工作，所以就在上班的路上，顺路把孩子送到学校去关着，这有点像他们周末外出度假时，把狗送到寄养的地方。

几乎在任何团体中都存在等级关系。成年人在真实世界中形成的团体，一般来说，都存在某个共同目标，团体的领导者往往由最善于实现目标的人担任。学校就不一样，大多数情况下，学生内部形成的团体没有一个共同目标。但是，等级关系却不会缺席，所以孩子们的等级是凭空创造出来的。

校园生活的真正问题是空虚。除非成年人意识到这一点，否则无法解决这个问题。

2 黑客与画家（黑客也是创造者，与画家、建筑师、作家一样）

黑客与画家的共同之处，在于他们都是创作者。与作曲家、建筑师、作家一样，黑客和画家都是试图创作出优秀的作品。他们本质上都不是在做研究，虽然在创作过程中，他们可能会发现一些新技术。

与其说优秀的软件设计师是工程师，还不如说是建筑师。建筑学和工程学之间的区别并不是很严格的，但就是存在区别。这表现在“做什么”和“怎么做”：建筑师决定做什么，工程师想出怎么做。 ？？？

创造优美事物的方式往往不是从头做起，而是在现有成果的基础上做一些小小的调整，或者将已有的观点用比较新的方式组合起来。这种类型的工作很难用研究性的论文表达。

唯一有效的外部考核就是时间。经过岁月的洗礼，优美的东西生存发展的机会更大，丑陋的东西往往会被淘汰。不幸的是，这种考核需要的时间可能比一个人的生命还要长。

※※在读研究生期间，我潜意识里一直有一种很不舒服的感觉，觉得自己应该多学一点理论，不应该期末考试结束还不到三个星期，就把所有东西忘得一干二净，那样真是不可饶恕。现在，我意识到自己错了。黑客搞懂“计算理论”的必要性，与画家搞懂颜料化学成分的必要性差不多大。我发现，黑客新想法的最佳来源，并非那些名字里有“计算机”三个字的理论领域，而是来自于其他创作领域。

我现在认为，大学里交给我的编程方法都是错的。你把整个程序想清楚的时间点，应该是在编写代码的同时，而不是在编写代码之前，这与作家、画家和建筑师的做法完全一样。明白这一点对软件设计有重大影响。它意味着，编程语言首要的特性应该是允许动态扩展。编程语言是用来帮助思考程序的，而不是用来表达你已经想好的程序。

真正竞争软件设计的战场是新兴领域的市场，这里还没有人建立过防御工事。只要你能做出大胆的设计，由一个人或一批人同时负责设计和实现产品，你就能在这里战胜大公司。

价格是由供给和需求共同决定的。好玩的软件的需求量，比不上解决客户麻烦问题的软件的需求量。在小剧场里演出的酬劳，比不上穿着卡通大猩猩服装、在展览会上为厂商站台的酬劳。写小说的回报比不上写广告文案的回报。开发编程语言的收入，比不上把某些公司老掉牙的数据库连上服务器的收入。

因为如果你不爱一件事，你不可能把它做得真正优秀，要是你很热爱编程，你就不可避免地会开发你自己的项目。

同样地，黑客可以通过观看优秀的程序学会编程，不是看它们的执行结果，而是看它们的源代码。开源运动最鲜为人知的优点之一，就是使得学习编程变得更容易了。

你不能盼望先有一个完美的规格设计，然后再动手编程，这样想是不现实的。如果你预先承认规格设计是不完美的，在编程的时候，就可以根据需要当场修改规格，最终会有一个更好的结果。

优秀的软件也要求对美的狂热追求。如果你查看优秀软件的内部，就会发现那些预料中没有人会看见的部分也是优美的。

如果黑客只是一个负责实现领导意志的技术工人，职责就是根据规格说明书写出代码，那么他其实与一个挖水沟的工人是一样的，从这头挖到那头，仅此而已。

如果一个代码块由三四个人共同开发，就没有人真正“拥有”这块代码。最终，它就会变得像一个公用杂物间，没人管理，又脏又乱，到处堆满了冗余代码。

事实表明，从他人的角度思考问题正是成功的奥秘所在。“换位思考”并不就意味着你要做自我牺牲。实际上，这是完全不同的两回事。了解别人对于事物的看法，并不代表你为他的利益服务。

《计算机程序的结构与解释》卷首语：程序写出来是给人看的，附带能在机器上运行。

在达·芬奇的年代，绘画并不是一件很酷的事情，达·芬奇用自己的工作推动绘画成为一种伟大的表达方式。同样，编程到底能够有多酷，取决于我们能够用这种新媒介做出怎样的工作。

3 不能说的话（如果你的想法是社会无法容忍的，你怎么办？）

历史的常态似乎就是，任何一个年代的人们，都会对一些荒谬的东西深信不疑。他们的信念还很坚定，只要有人稍微表示一点怀疑，就会惹来大麻烦。

我们这个时代是否有所不同？只要读过一点历史，你就知道答案几乎确定无疑，就是“没有不同”。即使有那么一丝微小的可能，有史以来第一次，我们这个时代的所有信念都是正确的，那也是出于惊人的巧合，而不是因为我们真找到了正确的方向。

你是一个随大流的人吗？

与历史上别的年代一样，我们的思想几乎肯定也是一张有错误的地图。如果你也犯下与别人一样的错误，那么这个错误不太可能完全来自于你自己。

真话

我猜想，最令人暴跳如雷的言论，就是被认为说出了真相的言论。

异端邪说

历史上的每一个年代，都会给“异端邪说”贴上一些标签，目的是在人们开始思考它们是否为真之前就把它们封杀。

我们可以自以为是地相信，当代人比古人更聪明、更高尚。但是，了解的历史越多，就越明白事实并非如此。古人与我们是一样的人，他们既不是更勇敢，也不是更野蛮，而是像我们一样通情达理的普通人。不管他们产生怎样的想法，都是正常人产生的想法。

不管实际上是否有思想斗争，斗争总是会以思想斗争的形式表现出来。正如刚刚过气的时尚并非一点儿也不时尚，失败一方的思想其实也并非一无是处。

优秀作品往往来自于其他人忽视的想法，而最被忽视的想法就是那些被禁止的思想观点。

做一个异端是有回报的，不仅是在科学领域，在任何有竞争的地方，只要你能看到别人看不到或不敢看的东西，你就有很大优势。

“守口如瓶”的真正缺点在于，你从此无法享受讨论带来的好处了。讨论一个观点会产生更多的观点，不讨论就什么观点也没有。所以，如果可能的话，你最好找一些信得过的知己，只与他们畅所欲言、无所不谈。这样不仅可以获得新观点，还可以用来选择朋友。

如果你的思想很保守，你自己不会知道，而且你很可能还会持有相反的看法。请记住，所谓“流行”（传统观念也是一种流行），本质上就是自己看不见自己的样子。否则就不会有流行了。

时间就是一种产生距离的简单方法，实际上，新的流行让旧的流行更容易被观察到，因为对比之下，旧的流行会显得很荒唐。从钟摆波动的一端望去，上一个周期的端点就显得特别遥远。

小时候，每个人都会鼓励你不断成长，变成一个心智成熟、不再耍小孩子脾气的人。但是，很少有人鼓励你继续成长，变成一个怀疑和抵制社会错误潮流的人。

如果自己就是潮水的一部分，怎么能看见潮流的方向呢？你只能永远保持质疑。问自己，什么话是我不能说的？为什么？

4 良好的坏习惯（与其他美国人一样，黑客的成功秘诀就是打破常规）

只有深入了解当前的技术，黑客才能构想下一代技术。知识产权的拥有者也许会说，不，谢谢，我们不需要你的帮助，我们自己就能开发下一代技术。他们错了，在计算机工业的历史上，新技术往往是由外部人员开发的，而且所占的比例可能要高于内部人员。

黑客的这种自行其是的特点，很大程度上说明了，为什么不管是出色的工作还是糟透了的工作，黑客都用同一个词（terrific）形容。如果他们做出来一个东西，他们自己总是无法百分百确定那到底是什么东西，有可能完全没用。但是只要那些出错的地方还算正常，那么就是一个信号，表明这个东西还有希望。

这就是为什么黑客感到担忧。政府侵犯公民自由，从表面上看，并不会让程序员的代码质量下降。它只是逐渐地导致一个错误观点占上风的世界。黑客对于公民自由是非常敏感的，因为这对他们至关重要。他们远远地就能感到极权主义的威胁，好比动物能够感知即将来临的暴风雨。

那些占据高位、本能地想要约束黑客、强迫黑客服从的人们，请小心你们的要求，因为你们真有可能成为千古罪人。

很自负的人必须培养出敏锐的感觉，及时发现周围情势的变化，知道怎样才能脱身。

5 另一条路（互联网软件是微机诞生后的最大机会）

大多数人，在大多数时候，总是选择最省事的做法。

“你的电脑”这个概念正慢慢成为过去时，取而代之的是“你的数据”。你应该可以从任何电脑上获取你的数据。或者更准确地说，在任何终端设备上获取你的数据，终端设备不一定是电脑。

互联网应用程序能够同时被多人使用，所以非常适合团队协作性的工作。

对于用户来水，使用互联网软件不会有太大的压力。我觉得，大多数Windows用户使用桌面软件的时候都会感到紧张，会有相当大的心理压力。施放这种压力，对你的产品将是一种巨大的推动。

但是，硬件需要考虑的地方，不仅仅在于怎么才能避免出问题，还在于怎样才能最大地发挥它们的作用。

对于互联网软件，没人规定只能使用某些语言开发，因为所有的硬件都控制在你手里，你想要用什么语言，就能用什么语言。不同的语言适合不同的人物，你应该根据不同的场合，挑选最合适的工具。尤其是在竞争者存在的情况下，“可以这样做”就变成了“必须这样做”，因为如果你不利用语言的优势，那就会听任对手超过你。

互联网软件的发布规则是：它运行不了，你就无法发布；一旦它能运行了，你就可以立刻发布。

不要被微软吓到。你能做到它做不到的事情，正如它能做到你做不到的事情一样。开发互联网软件不需要得到任何人的许可，没有任何能够阻止你。你不需要去申请许可证，不需要在零售店的货架上谋得一席之地，也不需要卑躬屈膝地求人家，将你的软件与操作系统捆绑在一起。你能够通过浏览器发布软件，没有人能在你和浏览网站的用户之间插上一脚。

6 如何创造财富（致富的最好方法就是为社会创造财富。创造财富的最好方法就是创业）

从经济学观点看，你可以把创业想象成一个圧缩过程，你的所有工作年份被压缩成了短短几年。你不再是低强度地工作四十年，而是以极限强度工作四年。在高技术领域，这种压缩的回报尤其丰厚，工作效率越高，额外报仇就越高。

通过创造有价值的东西而致富，这种方法的优势不仅仅在于它是合法的，还在于它更简单。你只需要做出别人需要的东西就可以了。

财富才是你的目标，金钱不是。但是，如果财富真的这么重要，为什么大家都把挣钱挂在嘴边呢？部分原因是，金钱是财富的一种简便的表达方式；金钱有点像流动的财富，两者往往可以相互转化，但是，他们确实不是同样的东西，除非你打算伪造货币。

我们这个世界，你向下沉沦或者向上奋进都取决于你自己，不能把原因推给外界。许许多多不创造任何财富的人——比如本科生、记者和政客——一听到最富有的5%人口占有全社会一半以上的财富，往往会认定这是不公平的。一个有经验的程序员很可能也认为这是不公平的。因为最顶尖的5%的程序员写出了全世界99%的优秀软件。

公司的一切行为的目的都是盈利，从而生存下去。创造财富是大多数公司盈利的手段。公司的业务高度专业化，掩盖了它们都是在创造财富的这种相似性，你不要觉得只有制造业公司在创造财富。

公司不过是一群人在一起工作，共同做出某种人们需要的东西。真正重要的是做出人们需要的东西，而不是加入某个公司。对于大多数人来说，最好的选择可能是为某个现存的公司打工。但是，理解这种行为的真正含义对你没有什么坏处。工作就是在一个组织中，与许多人共同合作，做出某种人们需要的东西。

要致富，你需要两样东西：可测量性和可放大性。你的职位产生业绩，应该是可测量的，否则你做得再多，也不会得到更多的报酬。此外，你还必须有可放大性，也就是说你做出的决定能够产生巨大的效应。

理想情况下，你与其他愿意更努力工作的人一起组成一个团队，共同谋取更高的回报。因为创业公司的团队往往是自发形成的，许多有抱负的创始人彼此之间早就相识（至少听说过对方），所以他们对彼此贡献的评估要比一般的小团体更准确。创业公司不仅仅是十个人的团队，而且是十个同类人的团队。

创业公司为每个人提供了一条途径，同时获得可测量性和可放大性。因为创业公司是小团队，所以具备可测量性。因为创业公司通过发明新技术盈利，所以具备可放大性。

小团队天生就适合解决技术难题。技术的发展是非常快的，今天很有价值的技术，几年后可能就会丧失价值。小团队在如今这个时代可谓如鱼得水，因为他们不受官僚主义和繁琐管理制度的拖累。而且，技术的突破往往来自非常规的方法，小团队就较少受到常规方法的约束。

创业是有一些潜规则的，其中一条就是很多事情由不得你。比如，你无法决定到底付出多少。另一条潜规则是，创业的付出与回报虽然总体上是成比例的，但是在个体上是不成比例的。

创业公司不像能经受打击的黑熊，也不像有盔甲保护的螃蟹，而是像蚊子一样，不带有任何防御，就是为了达到一个目的而活着。蚊子唯一的防御就是，作为一个物种，它们的数量极多，但是作为个体，却极难生存。

用户数量也许不是最好的测量指标，但应该也相差不远了。买家关心它，收入依赖它，竞争对手恐惧它，记者和潜在用户则是被它打动。无论你的技术水平有多高，用户数量都比你自己的判断更能准确反映哪些问题应该优先解决。

缓慢工作的后果并不仅仅是延迟了技术革新，而且很可能会扼杀技术革新。只有在快速获得巨大利益的激励下，你才会去挑战那些困难的问题，否则你根本不愿意去碰它们。开发新技术是非常痛苦的经历，正如爱迪生所说，百分之一的灵感加上百分之九十九的汗水。没有财富的激励，就不会有人愿意去做技术革新。工程师愿意接受普通薪水去做一些诱人的项目，而与日常生活关系更密切的技术革新只能由创业者发明。

一旦自己的财产有了保证，那些想致富的人就会愿意去创造财富，而不是偷窃。

7 关注贫富分化（“收入分配不平等”的危害，会不会没有完美想的那样严重？）

8 防止垃圾邮件的一种方法（不久前，许多专家还认为无法有效地过滤垃圾邮件。本文改变了他们的想法）

9 设计者的品味（如何做出优秀的东西？）

10 编程语言解析（什么是编程语言？为什么它们现在很热门？）

11 一百年后的编程语言（一百年后，人类怎样编程？为什么不从现在开始就这样编程呢？）

12 拒绝平庸（别忘了你的对手与你一样，能用任何想用的语言编写互联网软件）

13 书呆子的复仇（在高科技行业，只有失败者采用“业界最佳实践”）

14 梦寐以求的编程语言（一种好的编程语言，是让黑客可以随心所欲使用的语言）

15 设计与研究（研究必须是“新”的，而设计必须是“好”的）