

有 0 位同学正在浏览当前页面

大脑（记忆）运作原理

🔖 (/favorites?post_id=509)

为什么要采纳这样的学习方法呢？

一般人学习之所以低效，是因为不了解自己的大脑怎么运作。一旦你开始了解自己的大脑是怎么运作的，很快的，你就会发现学习是有套路的，而且你可以利用这套方法，大幅拉升自己的学习初速度。

在这里我先告诉各位五个结论：

- 大脑并不擅长思考，而且大脑的思考是很缓慢的
- 多数的思考，并不是真的思考，而是调用过去记忆所组成的结果
- 人是利用已知的事务理解新的事物，但「理解」其实是「记忆」
- 没有重复的练习，不可能精通任何脑力活
- 题海战术以及填鸭教育，有时是必须的

1. 大脑并不善于思考

在这社会上我们最常嘲讽的一个现象：「大多数人是不用脑子思考的」。其实这真是事实！

你仔细想想，其实大脑真是用来思考的吗？如果你叫大脑随便做一题演算，其实大脑的演算，往往是比我们现在所发明的计算机来说，效率是极其低的。做个 $7 * 8$ 的数学还行，但要是改个 $177*288$ 的快速演算。就瞬间就当机了。

蜡烛、火柴、图钉

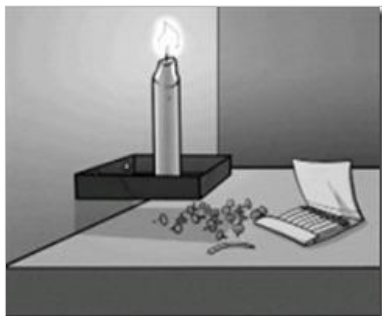
在这里，我举一个「大脑其实不善于思考」的例子。

一个空屋子里有一支蜡烛，一些火柴，和一盒图钉。目标是让点燃的蜡烛离地五英尺高，你已经尝试把蜡烛底部沾上蜡液，但还是沾不到墙上，怎样才不用手扶，让点燃的蜡烛离地五英尺高？

这一个题目，正常一般人在看到题目后，很少能在 20 分钟内给出解答答案。

但是如果你把这个题目「具象化」，也就是真的生出这些设备，放在眼前。

你就会发现这道题目的答案其实并不难。你只要把图钉钉在墙壁上，再把蜡烛黏在盒子里，就完成了这个任务。



([https://s3-ap-northeast-1.amazonaws.com/ontrackapp-](https://s3-ap-northeast-1.amazonaws.com/ontrackapp-production/Zqh3iOWmQT6Nx8MMX8s_20130803_candle-problem_02_thumb.jpg)

[production/Zqh3iOWmQT6Nx8MMX8s_20130803_candle-problem_02_thumb.jpg](https://s3-ap-northeast-1.amazonaws.com/ontrackapp-production/Zqh3iOWmQT6Nx8MMX8s_20130803_candle-problem_02_thumb.jpg))

大脑的「思考」特性

这个例子解释了「思考」的几个特性。

- 首先，大脑的思考是很缓慢的。
- 接着，思考是很费力的。大脑很难凭空想像出这个场景并运算出解答。甚至可能「完全答不出来」。
- 但是如果把大脑接上视觉系统与触觉系统。因为视觉系统与触觉系统进行了可靠的回传，大脑实质上是调用了其他地区可用的资源做了运算。就能迅速得出答案。

那么，既然思考那么费力。我们平时是怎么样不费工夫的做出日常生活中的各样决策？

习惯

答案是：习惯。

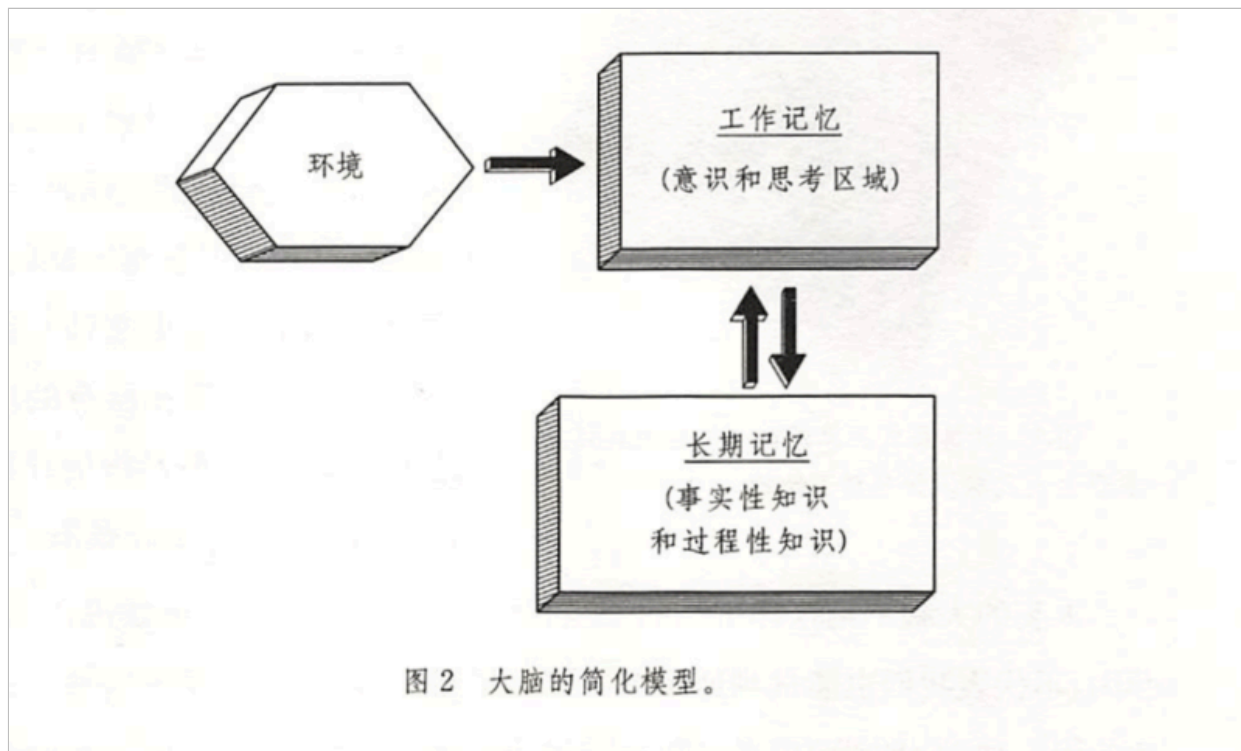
「习惯」就是「我们做过某件事的记忆回路」，大脑调用「过去的记忆」，让身体自动做出判断。

所以，在这里，我们要引出今天要介绍的第二条认知学事实：

大部分人做的决策，其实真不是基于大脑所做的思考，他们是「记忆」组成的结果

2. 多数的思考，并不是真的思考，而是调用过去记忆所组成的结果

大脑的运作原理是这样的：



(https://s3-ap-northeast-1.amazonaws.com/ontrackapp-production/fGpKtc9fRR9gQg5YQPza_%E8%9E%A2%E5%B9%95%E5%BF%AB%E7%85%A7%202016-12-30%20%E4%B8%8B%E5%8D%882.17.37.png)

接收到环境刺激 => 然后把决策放到工作记忆上=> 熟练之后烧到长期记忆中（事实性知识、过程性知识）。

- 工作记忆就是我们当前正在意识、思考的「工作区域」。
- （以计算机比喻，就是电脑的内存。容量小，资料存在时间短，重开机就不见了。）
- 长期记忆就是我们长久以来储存的事实性知识、经验。
- （以计算机比喻，就是电脑的硬盘。容量大，资料存在时间长，可以长期复用。）

而长期以来，我们日常遇到的大量决策，事实上是调用了长期记忆（经验以及不变的科学事实），自动完成。

而所谓的解题与思考，是复用了短期记忆以及长期记忆而成。

比如说以 $177 * 288$ 这个例子

- $7 * 8$ 是长期记忆，是我们小时候背的九九乘法。
- 接下来我们要算 $170 * 280$ 。接下来
- 好了。你知道这有多难了。

人类几乎很难凭空展开这个算式。

这是因为人类的大脑工作记忆中只能暂时存 7 ± 2 个结果（对人类无意义的结果）。人类事实上很少在思考，更多的是调用「记忆」在做决策。

3. 人是利用已知的事务理解新的事物，但「理解」其实是「记忆」

人是利用已知的事务理解新的事物，但「理解」其实是「记忆」。

「理解」其实不是一个调用大脑思考的过程，而是一个匹配「记忆」的过程。

很多行业为什么不喜欢招小白，事实上就是因为小白经验过少，缺乏太多相关记忆，可以直接匹配学习。

又或者是为什么一般人难以上手编程，而且对于学习编程，感受到痛苦。

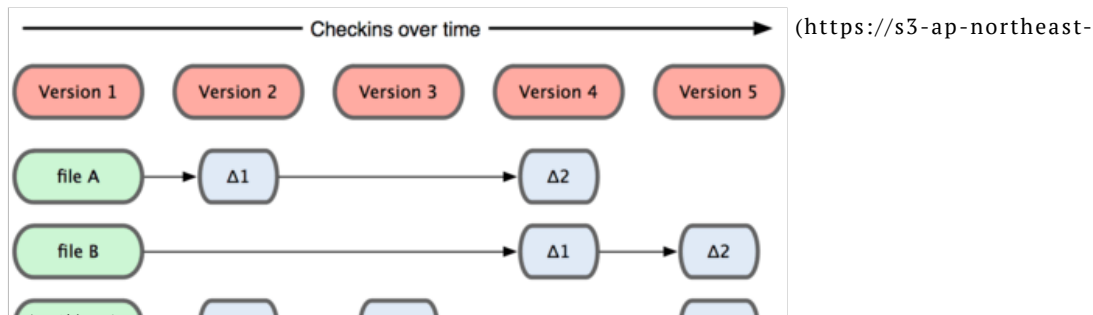
这事实上就是因为「编程」里面的知识，跟过往几乎所有的生活环境运作原理几乎是不匹配的，所以造成「无法理解」，而大量调用大脑资源匹配、思考，结果却一无所返的情况下，造成意志力崩溃。

很多人学习编程事实上是被一堆所谓枯燥的「基础知识」，所吓跑的。

老手觉得「基础知识」是很重要的，但「基础知识」恰恰对许多新手来说，是「无法理解的」。

编程书反例：Git 如何运作

比如说，我引述一本所谓程序员界的 Git 初学指南，来谈谈：「Git 如何运作」。



VIP入学手册 (/courses/28/syllabus) 搜索本课程教材

概览 (/courses/28) 教材 (/courses/28/syllabus) 1.amazonaws.com/ontrackapp-production/s7OFC1AaTa2RXHvTEvfO-18333fig0104-tn.png)

作业 (/courses/28/assignments)

FAQ (/courses/28/faqs)

动态 (/courses/28/activities)

积分榜 (/courses/28/leaderboard)

『那么，简单地讲，Git是一个什么样的系统？这一章节是非常重要的。若读者了解Git的本质以及运作的基础，那么使用起来就会很轻松且有效率。在学习之前，试着忘记以前所知道的其它版本控制系统，如：Subversion 及 Perforce。这将会帮助读者使用此工具时发生不必要的误会。Git储存资料及运作它们的方式远异于其它系统，即使它们的使用者介面是很相似的。了解这些差异会帮助读者更准确的使用此工具。』

Git与其它版本控制系统（包含Subversion以及与其相关的）的差别是如何处理资料的方式。一般来说，大部份其它系统记录资讯是一连串档案更动的内容。这些系统（CVS、Subversion、Perforce、Bazaar等等）储存一组基本的档案以及对应这些档案随时间递增的更动资料。

Git并不以此种方式储存资料。而是将其视为小型档案系统的一组快照(Snapshot)。每一次读者提交更新时、或者储存目前专案的状态到Git时。基本上它为当时的资料做一组快照并记录参考到该快照的参考点。为了讲求效率，只要档案没有变更，Git不会再度储存该档案，而是记录到前一次的相同档案的连结。Git的工作方式如图1-5所示。』

「若读者了解Git的本质以及运作的基础，那么使用起来就会很轻松且有效率」，听起来是多么讽刺啊。

绝大多数需要学 Git 如何入门的人，是一无所知的小白。Git / Subversion / Snapshot / 参考点，这几个名词，对一般人来说是无意义的。不只是一般人，甚至是一辈子都在使用 FTP 部署代码的的程序员来说，他也不了解你在说什么。

正确比喻：时光机

所以呢，一般要怎么让「读者了解Git的本质以及运作的基础，使用起来很轻松且有效率」。

1. 告诉他 Git 是个时光机
2. 你可以用一套图形化的工具操作这个时光机，任意回到希望回溯的时间点。之后你可以在任意时刻检查你之前所写的代码，并检查变化。不仅是开发团队，甚至是很多写作团队，也开始利用这套工具实行协作。
3. 因为绝大多数的人看过小叮当或者是科幻电影，所以他能够理解时光机是什么，也知道时光机能带来的好处。进而愿意接触以及使用这个工具增进工作效率。甚至可能在听完我讲这个例子时，就已经对 Git

这套工具产生兴趣，实际去使用了。

而等到使用者理解了Git 可以帮他做什么之后，且想控制更多细节时，就会愿意使用命令列，操作更高级的命令与效果，进而去研究Git 的底层运作，学到更多奇技淫巧，从而「使用起来更轻松且有效率」。

所以之前这份 git 入门教程到底是写给谁看的？我只能这样说，我认为这不是写给入门者看的，即便这章叫做「Git 基础要点」。这就是一般编程书的坑。

我们话题扯得有点远了。总之呢？你得记住这样一个结论：

人类只能利用已知的事物理解新的事物

而这是许多教育界圣经，如教育七律，反覆提及的教学要点。

人类的「理解」，实质上是一道不断在旧有记忆上不断叠加累积的过程。

4. 没有重复的练习，不可能精通任何脑力活

如果一个人，带球的同时还要思考踢球的角度和速度，不太可能成为一个优秀的足球选手。像这样的低层次过程必须不假思索，才能给更高层次的过程，比如战术策略提供足够空间。

正因为我们的工作记忆之狭小，如果你将大量需要调用的资源放在工作记忆，那么大脑就会瞬间寸步难行。就如同这个足球选手来说，如果他要一边思考踢球角度和速度，那么下场不但不只是「无法射门」，甚至可能是「跌倒」。多半优秀的足球选手进行射门，往往是凭「直觉」判断，「感觉」角度对了，场上有空档，直接踢出一个漂亮弧度的球，射门成功。

这个直觉，往往就是 **深焊在肌肉里面的长期记忆**。

而一般人所谓的学习，事实上是在将新东西存在短期记忆中，进行「理解」的连结。再透过反覆的练习，往下放置在长期记忆中。

5. 题海战术以及填鸭教育，有时是必须的

在过去我们的应试教育中，因为我们大量的被灌输填鸭教育以及题海，以至于我们痛恨「题海战术」以及「填鸭教育」。所以，往往对于「肌肉的记忆」练习术这件事十分不苟同。

甚至是，因为我们对于「题海战术」过于痛恨，甚至得到了相反的结论：「我们未来教育学生，必须使用理解型教育」这个甚至看似正确，但事实上是谬误的决策。

注意啊。在前面我讲了「理解」实际上是「记忆」的相关连结。

如果你的大脑，从来未曾存在相关记忆，那么又如何「理解」。又比如说，你如何让幼儿理解 $7 * 8 = 56$ ；中学生理解 $e = mc^2$ ；想学编程的大学生，理解 git 的 repositroy 机制。

答案是：你不能。

因为事实上他们就没有这样的相关记忆可以去做链结。所以你只能让他们硬背，直接先锁在记忆区里面。等待将来更有意义的相关材料，进一步的将这些硬背的东西，锁到更深的地方。

6. 右脑模式开始探索整体框架

如果「编程」是你这辈子从未碰触过的学问，那么按照大脑最容易入门的方式，其实就是：

不要强求自己用眼睛，甚至不用大脑去理解，把左脑模式关掉

尽量的摸索一个外围框架，在大脑深处种下记忆点。（比如说你只要知道「输入什么」，然后可以「得到若干输出」就好了。（如果你连什么是有效的外围框架都不知道。不如直接付费去上新手班，从教练身上学一个一个外围高频套路，可以少掉很多坑）

透过肌肉的练习，把这些记忆点种到肌肉里面。（打错字不再犯，无法理解但常用的都先背起来）

形成了一个防摔记忆层后，再用左脑去分析自己未来想知道更多的东西

一层一层的用「已知解释已知」。


当然，在这当中要保持「开心」、「有成就感」，不妨完成一个小作品后，就扔上网给那些也不懂编程的朋友看一下，得到他们的赞美。唯有充满成就感的学习，才是初学状态最重要的。有成就感的进步，才容易形成习惯回路。

这一路写下来，相信你开始可以理解，为什么社会上「政治正确」的这些「常识」：

- 学习编程必须要有天赋
- 必须在学习编程中「看」懂每一个步骤
- 学习必须要扎实，要从技术基础原理学起
- 学习当中禁止使用捷径，如复制代码

是有多么毒的吧！如果按照社会上这套「常识思路」，「编程初学者从入门到放弃」根本就是 99% 可以预期的结果！

对本页内容的感受如何？

 我要吐槽



So easy



还OK



崩溃了

[← 上一页 \(/posts/508\)](/posts/508)


🔍 可以使用   键进行翻页

[下一页 → \(/posts/510\)](/posts/510)

全栈营

课程资源

关于我们

 在线客服（非技术答疑，工作日10:00-19:00）

课程介绍

学习中心

公司介绍

[\(/pages/course_intro\)](/pages/course_intro)

[\(/dashboard\)](/dashboard)

[\(/pages/about\)](/pages/about)

教学团队

帮助文档

常见问题

[\(/pages/teachers\)](/pages/teachers) <http://docs.qzy.camp> [\(/pages/faq\)](/pages/faq)

学员心得

交流论坛

联系方式

[\(/pages/students\)](/pages/students) <http://forum.qzy.camp> [\(/pages/contact\)](/pages/contact)



新生大学 - 软件学院是李笑来对未来世界的实验计划，旨在改变中国的计算机教育。
[\(http://www.xinshengdaxue.com/\)](http://www.xinshengdaxue.com/)

