推荐序

在创新工场内，学镛是一个很特殊的人，他的职位是首席布道 / 架构师，在中国能够担任这样职位的人有如凤毛麟角，因为架构师（Architect）必须有很多年的软件开发实务经验，而布道师（Evangelist）必须熟知新技术且热爱宣传技术，通过写作、宣讲等方式推广技术。两者结合，且做得相当好，确实不容易。

在 IT 方面，学镛是我认识最执着且对技术的深度与广度都能兼顾的人。他对软件技术的热爱是发自内心的，且似乎总能从学习新技术的过程中得到乐趣。他写过许多技术文章与书，参与过许多软件的开发，讲过许多技术课。现在他在创新工场，对我们的创业团队进行技术上的指导以及担任投资项目的技术评审。

尽管学镛是个专业的人，但他另一个很强的特点是：擅长把复杂的技术用简单清楚的方式描述出来，这本《编程 ING：人人都能学会程序设计》正是这样的一本书。要让“人人”都能学会程序设计，这是一个很难又相当有价值的目标。通过这本书，学镛确实做到了。

人人都能学会程序设计，并不是说人人都应该以软件工程师为职业。现在社会高度电脑化，我们每天与手机、平板电脑等设备上的各种软件或网站为伍，如果我们能多懂一点软件相关的知识，甚至能写简单的程序解决一些生活上的小问题，这是多么棒的事！

这本书的风格非常像微博，一张图搭配一则短文，读这本书就像是读了三百多则图文并茂的微博。这本书也展现出学镛的 PPT 设计功力，每张图都是他自己精心绘制的。将概念图像化，对于学习的帮助很大。

我喜欢这本多彩多姿、深入浅出、走入群众的书。我相信你也会喜欢。

创新工场董事长兼首席执行官

自序

我做过很多不同类型的工作，包括大学讲师、培训师、软件工程师、架构师、技术图书编辑、译者、专栏作家，但其实这些工作都围绕着程序设计这个专业领域，因为我热爱程序设计。

程序设计既有趣，又有创造力，还能帮助提升日常工作效率。你只需要带着你的想法与一台计算机，就能开始进行程序设计，把脑海中的想法在计算机中实现出来。我有幸很早就体会到程序设计的迷人之处，从小学开始学习写程序，至今 29 年，依然喜欢。可惜的是，像我这样幸运的人毕竟是少数，有许多人对于程序设计感兴趣却又不得其门而入。

多年以来，我一直想写一本程序设计入门书，以帮助程序设计初学者。为此我倾注了相当大的心力。现在，我的目标终于达成，成果就是你手上的这本书。对于初学者来说，死板的理论与生硬的说教都是禁忌，只会让初学者打退堂鼓。面对初学者，我必须发挥创意，让这本书的内容深入浅出。除此之外，趣味性与实用性也是必要的，可让初学者保持学习的动力。

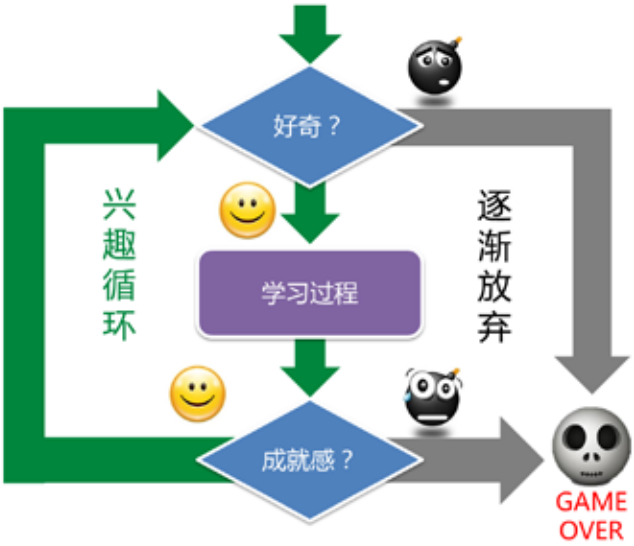
拿着这本书，用一个周末假期的时间仔细阅读并动手操作，你很可能会发现，原来程序设计这事儿可以这么有趣，这么吸引人。接下来你或许要担心上瘾了！

蔡学镛

于北京中关村

前言

学习编程的心理准备



待在兴趣循环内，别让你的学习 Game Over！

想要有良好的学习成果，必须进入兴趣循环。这个循环由好奇心、学习过程，以及成就感组成。一开始由好奇心触发学习动机，接下来展开学习，学习后产生成就感，而对更深入的内容感到好奇，于是继续学习。一旦没了好奇心或成就感，很可能就会放弃。

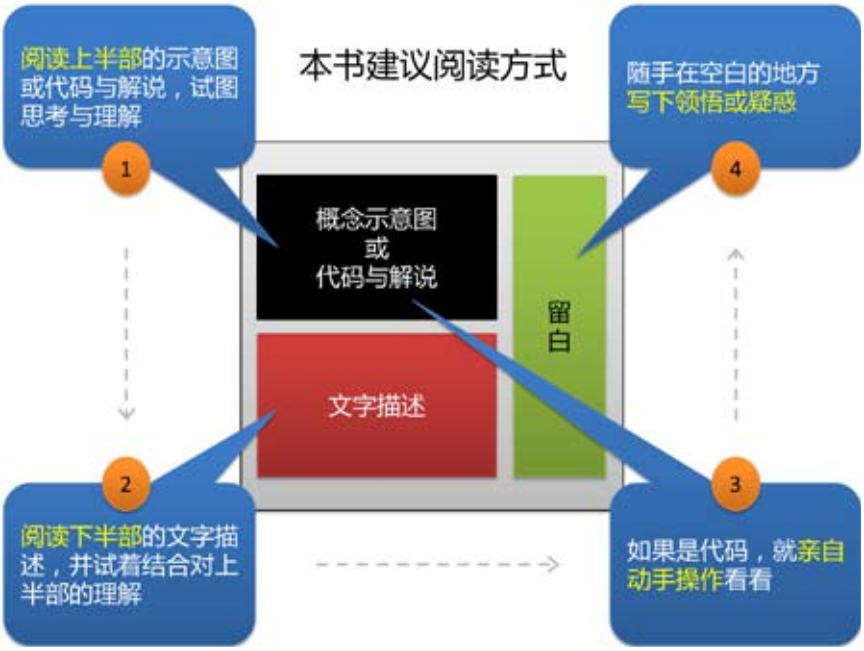
成就感是一种心理状态，与挫折感相反。想获取成就感，就需要有好的学习成果。请务必坚持学习，直到下一次成就感产生。

除了成就感，好奇心也可以刺激学习。不妨带着一丝疑惑进行学习和探索，直到真相大白的那一刻，那是一种豁然开朗的喜悦。

如何待在兴趣循环内？你需要成就感、好奇心、目标与奖赏。

前面提到成就感与好奇心是学习的两大关键，不过成就感与好奇心可不是说来就来的，但你可以通过一些手段激发你的成就感与好奇心。

1. 给自己定好许多可行的短期目标。如果你不知道该定怎样的目标，可以参考本书每篇一开始列出来的学习目标，每次达成目标，就勾选该目标前的方框。当你很肯定地勾选时，表示目标达成，你的内心会出现一丝成就感。
2. 你可以在达到一定的学习目标之后，就犒赏自己。例如完成四个学习目标，就奖赏自己奢侈地大吃一顿（如果大吃一顿是你所热爱的）。对于奖赏的渴望，会让你的学习可以坚持得更久一点，学习过程也会更顺利一点。奖赏自己的时候，成就感会更明确。
3. 有了成就感，你就会想要继续挑战下一个目标。整体进入一个良性循环。
4. 好奇心会在你良性循环的学习过程中随时出现，比较难捉摸。请务必把握机会，在好奇心出现时，加强学习。

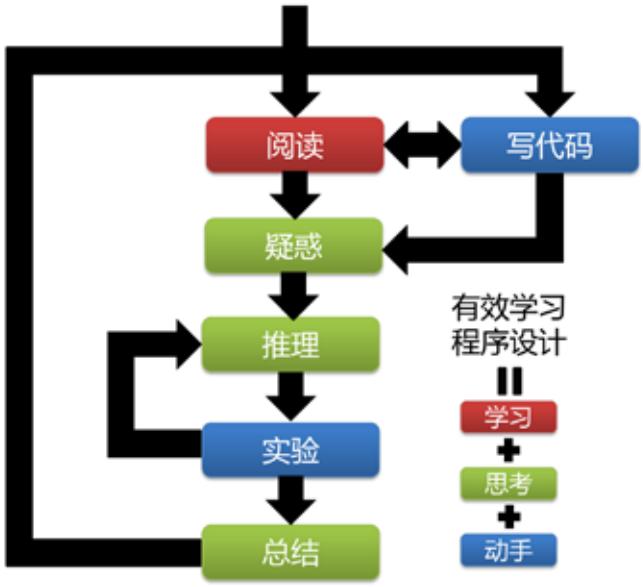


采用正确的方法，可以提高学习效率。

“书都看了，也都看懂了，但还是不会写代码”，很多人有这样的问题。学习效果不佳，通常是因为没有思考与动手所致。如果你保持思考与动手的习惯，并坚持一段时间，我保证学习成效会不错。

你必须一边阅读，一边思考，甚至质疑书中的内容。动手跟着书本实际操作，以加深印象。对于不清楚的部分，通过动手实验得到解答。把无法证实的疑惑，立刻记录下来，等待以后某天知识积累足够了而顿悟。

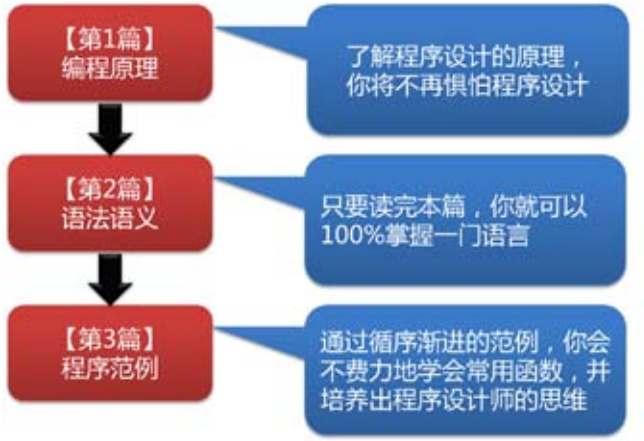
阅读本书每一页都可以采用图中描述的这四个步骤。首先阅读并思考上半页的精华，再看下半页文字描述的细节，接着动手操作实际领悟，这个过程中有任何问题与想法都要马上记录下来。



有效的学习 = 学习 + 思考 + 动手

学习程序设计与学习其他技术一样，不能光看书，必须从实际中学习，才有实际效果。所以请先准备好一台计算机，台式机或笔记本都可以。计算机上会自带一个操作系统（OS），最有可能是微软的 Windows，或苹果的 Mac OS X，也可能是其他操作系统。当然仅有计算机与操作系统依然不够，想做程序设计还必须有编译器或解释器，关于这方面，我在本书第 1 章中会说明。一开始不知道该写什么代码时，可以跟着本书的范例一起动手。

想学好程序设计，怀疑的态度与设计实验的能力也是相当重要的。多多怀疑某些事，然后推导出一番原理，最后做实验证实或推翻自己的想法。有这种好奇心与实验精神的人，才可能学好程序设计。

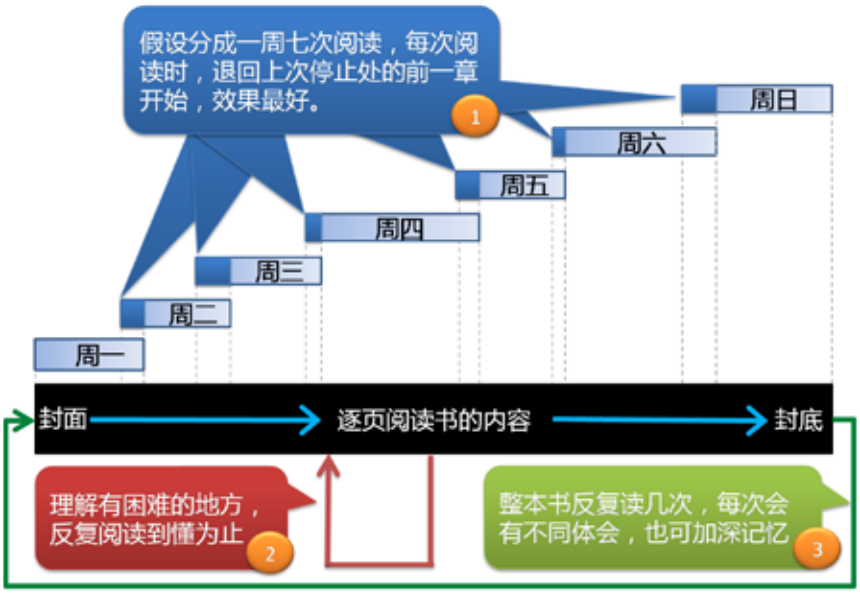


本书共有三篇，这三篇各具目的。

第 1 篇是编程原理，这会是你见过的最详细的程序设计概念解说。读完本篇，你就能了解程序设计的原理，有了大局观，你将不再惧怕程序设计。

第 2 篇是语法语义，完整地解说一个语言，没有遗漏。不可思议的是——只要学习完这么简短的一篇，你就能 100% 了解一门语言。接下来就可以迎接真正程序设计的挑战了。

第 3 篇是程序范例。前两篇有一些简单的操作，目的是让你熟悉语言个别元素，但第 3 篇会用更具体的范例讲解，有具体的需求，具体写代码。这些代码是逐渐递增功能的，所以学习坡度相当和缓。通过这些代码，你将会学习到许多常用函数，并培养出专业程序设计师一样的思维。

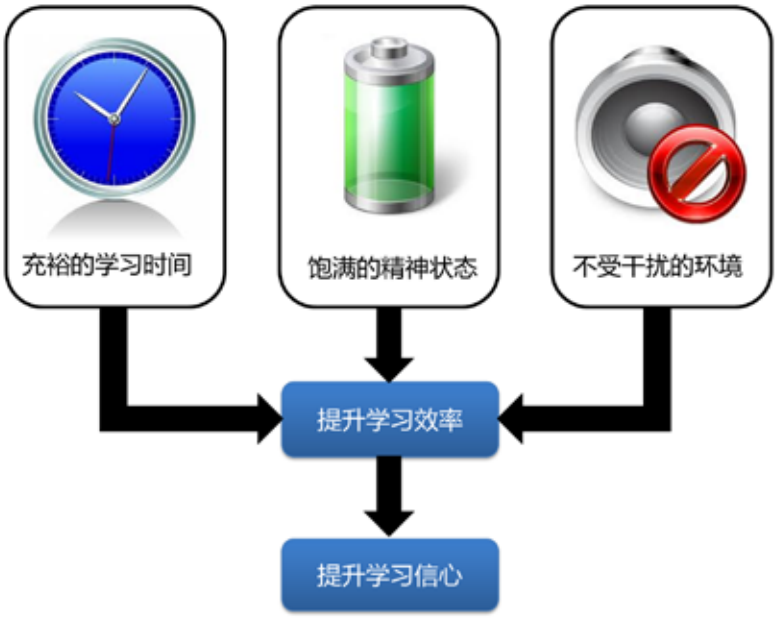


学习的次序：重叠、局部反复、整体反复

我们往往无法一口气读完一本书（尤其是要一边阅读，一边思考，一边动手操作），通常要分很多次才能阅读完毕。每次阅读时，我建议不要从上次停下来的部分开始，而应该倒退一两章开始。重叠的部分一方面可当做复习，另一方面也可以让自己进入上次的心理状态。另外，上次阅读停顿可能是因为学习效果开始大幅下降了，这种情况下重复阅读可以弥补上次学习时理解上的缺失。

本书内容前后有相当高的依赖性，如果某些概念没弄懂，对于后续的学习会形成障碍。所以我建议，对于理解有困难的章，要多读几次，直到懂为止，不要轻易跳过。

整本书读完之后，你还可以从头读第二次、第三次，每次都会有不同的收获，而你对程序设计的理解会越来越清晰。



时间、环境、精神都必须配合

学习时的时间、精神状态与环境都很关键。如果没有充裕的学习时间、饱满的精神状态与不受干扰的环境，学习效果不可能很好。更糟糕的是，这会给你一个假象：我不是学习程序设计的料。这种心理暗示的杀伤力很大。

周末假期睡眠充足，然后把手机等干扰物都关了，一整个下午和晚上关在房内读这本书，效果是最好的。

目录