

《老码识途》

图书基本信息

书名：《老码识途》

13位ISBN编号：9787121173820

10位ISBN编号：7121173824

出版时间：2012-8

出版社：电子工业出版社

作者：韩宏

页数：344

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《老码识途》

内容概要

《老"码"识途:从机器码到框架的系统观逆向修炼之路》以逆向反汇编为线索，自底向上，从探索者的角度，原生态地刻画了对系统机制的学习，以及相关问题的猜测、追踪和解决过程，展现了系统级思维方式的淬炼方法。该思维方式是架构师应具备的一种重要素质。《老"码"识途:从机器码到框架的系统观逆向修炼之路》内容涉及反汇编、底层调试、链接、加载、钩子、异常处理、测试驱动开发、对象模型和机制、线程类封装、跨平台技术、插件框架、设计模式、GUI框架设计等。

书籍目录

- 第1章 欲向码途问大道，锵锵bit是吾刀 1
 - 1.1 全局变量引发的故事 2
 - 1.1.1 剖析赋值语句机器码 2
 - 1.1.2 修改赋值语句机器码 6
 - 1.1.3 直接构建新的赋值语句 7
 - 1.1.4 小结 10
 - 1.2 理解指针和指针强制转换 11
 - 1.2.1 指针和它丢失的类型信息 11
 - 1.2.2 指针强制转换 13
 - 1.3 函数调用和局部变量 15
 - 1.3.1 计算指令中的跳转地址 15
 - 1.3.2 返回故乡的准备 16
 - 1.3.3 给函数传递参数 17
 - 1.3.4 函数获取参数 18
 - 1.3.5 局部变量 20
 - 1.3.6 返回故乡 20
 - 1.3.7 返回点什么 23
 - 1.3.8 扫尾工作 28
 - 1.3.9 调用惯例 30
 - 1.3.10 函数指针 31
 - 1.4 数组、结构体 34
 - 1.4.1 数组 34
 - 1.4.2 结构体 35
 - 1.5 无法沟通——对齐的错误 37
 - 1.5.1 结构体对齐 37
 - 1.5.2 无法沟通 41
 - 1.6 switch语句的思考 44
 - 1.6.1 switch机制探索 45
 - 1.6.2 switch语句一定比if-else语句快吗 50
 - 1.6.3 switch的再次优化 50
 - 1.7 关于其他高级语言要素的反汇编学习 54
 - 1.8 全局变量的疑问——重定位和程序结构 54
 - 1.8.1 独一无二的全局变量？ 54
 - 1.8.2 不变的地址和重定位 56
 - 1.8.3 动态链接库中的重定位 64
 - 1.9 汇编的学习之路——阅读RTL 68
 - 1.10 程序设置说明 76
- 习题1 76
- 第2章 庖丁解“码”：底层的力量与乐趣 79
 - 2.1 解密之hello world 80
 - 2.2 奇怪的死循环 83
 - 2.3 我们都犯过的错——指针的指针 85
 - 2.4 互通的障碍（跨语种调用） 87
 - 2.5 错误的目的地 90
 - 2.6 网络发送出错了 91
 - 2.7 为什么代码运行完毕却出错 93
 - 2.8 失效的管道 96

- 2.8.1 管道的力量 97
- 2.8.2 我要控制Telnet客户端 101
- 2.8.3 不是所有管道都可抽象等价 101
- 2.8.4 一动不动的48小时 103
- 2.9 异常世界历险记 112
 - 2.9.1 学习基础概念 112
 - 2.9.2 如何返回 113
 - 2.9.3 那些状态保存到哪里了 117
 - 2.9.4 意外的秘密 120
- 习题2 127
- 第3章 成长：与程序一起茁壮 131
 - 3.1 初写系统 132
 - 3.1.1 代码风格 132
 - 3.1.2 常量 133
 - 3.1.3 最简单的电话簿（1）：功能设计和相关库函数学习 134
 - 3.1.4 最简单的电话簿（2）：系统实现，分割函数 141
 - 3.1.5 空字符结尾串的警觉 143
 - 3.1.6 让程序更有组织性 144
 - 3.2 有序的世界：可测试与跟踪的系统 146
 - 3.2.1 电话簿扩展（1）：硬盘结构体数组 146
 - 3.2.2 指针的陷阱 148
 - 3.2.3 动态数组 149
 - 3.2.4 变化的压力与出路：重构、单元测试和日志 151
 - 3.2.5 电话簿扩展（2）：可测试的恩赐 155
 - 3.3 优雅的积木 155
 - 3.3.1 可复用硬盘数组 155
 - 3.3.2 分享它（1）：理解编译链接过程 161
 - 3.3.3 分享它（2）：我的丑陋链接器 167
 - 3.3.4 分享它（3）：静态链接库 173
 - 3.3.5 分享它（4）：动态链接库 175
 - 3.3.6 积木的艺术 178
 - 习题 3 182
- 第4章 让我们创造面向对象语言吧 185
 - 4.1 “封装”数据函数合一，陈仓暗度this传递 186
 - 4.1.1 那些讨厌的事 186
 - 4.1.2 像芯片一样工作（1）：数据合一 187
 - 4.1.3 像芯片一样工作（2）：行为与数据合一 188
 - 4.1.4 不想让你传递“自己” 189
 - 4.1.5 创造吧，新的语言 190
 - 4.1.6 是这样吗？我们需要证明 191
 - 4.2 太麻烦了，需要更简单的创造与销毁 194
 - 4.2.1 创造构造和析构函数 194
 - 4.2.2 构造中分配资源，析构中释放资源 197
 - 4.3 对比C语言的“对象”和面向对象 199
 - 4.4 体验封装的力量 201
 - 4.4.1 生死原点，整体资源管理 201
 - 4.4.2 文件流 203
 - 4.5 整体资源管理的爱恨 204
 - 4.5.1 扩展技巧：保证成对出现，巧妙的自动线程锁 204

- 4.5.2 美丽的幻影：不可靠的自动析构 205
- 4.5.3 隐藏的敌人：不请而至的析构和拷贝构造 206
- 4.6 封装之强化：内外之别，亲疏之分 209
 - 4.6.1 私有的诞生 209
 - 4.6.2 私有？阻止不了我 210
 - 4.6.3 理解继承的机制（1）：模型 211
 - 4.6.4 理解继承的机制（2）：在C语言中“玩”继承 214
 - 4.6.5 保护的诞生 218
- 4.7 “变”的烦恼与出路：创造虚函数 218
 - 4.7.1 “三变”之苦：格式化字符串 218
 - 4.7.2 函数指针，请带我走出不断修改的泥潭 220
 - 4.7.3 再进一步：做成对象 221
 - 4.7.4 我们需要性能更好的版本 223
 - 4.7.5 我们需要新语法，创造虚函数吧 225
 - 4.7.6 验证虚表机制（1）：反汇编分析 226
 - 4.7.7 验证虚表机制（2）：直接用虚表来调用虚函数 227
- 4.8 虚函数的那些事儿 227
 - 4.8.1 理解“=” 227
 - 4.8.2 纯虚函数，从dll导入对象 230
 - 4.8.3 C语言实现虚函数 231
 - 4.8.4 魂归何处：析构之“虚” 232
 - 4.8.5 理解运行期类型判断dynamic_cast 232
- 4.9 静态覆盖 235
- 4.10 静态与非静态成员函数的区别 235
- 4.11 遥远的风景：管窥.NET对象 235
- 习题4 236
- 第5章 底层与抽象的混沌：一个跨平台线程类的封装、错误与进化 239
 - 5.1 先学习多线程编程吧 240
 - 5.1.1 概念 240
 - 5.1.2 Windows下的线程接口 240
 - 5.1.3 第一个线程程序 242
 - 5.1.4 那些复杂的参数和bug 243
 - 5.2 简单、重用，让我们构造线程类吧 247
 - 5.2.1 无赖的尝试，原来是它——static 248
 - 5.2.2 可爱的virtual和可恨的this 249
 - 5.2.3 私有、保护、公有、只读、纯虚函数，一个都不能少 251
 - 5.2.4 析构中释放资源 252
 - 5.2.5 我们发现了一个设计模式 252
 - 5.2.6 我关心，你通知——我们的第二个设计模式 253
 - 5.3 跨平台的线程设计 255
 - 5.3.1 讨厌的Linux版本 255
 - 5.3.2 源代码跨平台技术 256
 - 5.3.3 跨平台的版本 257
 - 5.4 崩溃，哪里出错了 262
 - 5.4.1 寻找错误 262
 - 5.4.2 C++下整体资源管理的反思 265
 - 5.4.3 生生死死虚表误，剥离策略世界殊——重生 267
- 习题5 268
- 第6章 插件养成记 271

- 6.1 一个修改已有功能的实例 272
- 6.2 一个可以动态添加功能的简单实例 273
- 6.3 一个可以动态添加功能的复杂实例 276
- 6.4 从函数到插件对象 280
- 6.5 delete的灾难：谁的书 283
 - 6.5.1 释放内存的崩溃 283
 - 6.5.2 解决之道：新生活，各管各 288
- 习题6 291
- 第7章 天堂的阶梯 293
 - 7.1 遥望天堂，那些美丽与简洁我向往 294
 - 7.2 从最基础开始吧，SDK编写窗体程序 295
 - 7.2.1 hello window和基本原理 295
 - 7.2.2 来个复杂点的窗体程序 298
 - 7.3 构建我的GUI组件（1）：简单组件 300
 - 7.4 构建我的GUI组件（2）：天堂的机器码跳板 304
 - 7.4.1 调试，我要看清你 304
 - 7.4.2 我们的自定位代码 313
 - 7.4.3 自定位代码版Button类 314
 - 7.4.4 自定位代码版Form类 315
 - 7.4.5 为什么不错呢 316
 - 7.5 构建我的GUI组件（3）：更多的组件 317
 - 7.6 天堂阶梯，玩赏框架那如花散落的繁复与如索串珠的简洁之美 319
 - 7.7 构建我的GUI组件（4）：我的天堂 326
 - 7.8 他们的天堂 330
 - 习题7 332

章节摘录

版权页：插图：那么，登记窗体句柄和处理函数的对应关系有两种方法。在窗体描述结构WNDCLASS中指出。窗体千差万别，如果让系统构造一个窗体，需要给系统一份窗体图纸。该图纸就是WNDCLASS结构体，其中有两个重要的成员变量，一个是DM7—2的第13行的lpszClassName，标识这份图纸的名字，例中给定名字为“testwin”。后面构造窗体时，就用这个名字告知系统我们想用哪份图纸构造，第15行CreateWindow构造窗体，其第1个参数就是“testwin”，指出了图纸的名字。要用这份图纸自然需要将它注册给系统，所以第14行RegisterClass完成了该工作。WNDCLASS结构体另一个成员是第12行的lpfnWndProc，指出了窗体处理函数的地址，指示的函数就是第1~9行的函数WndProc()。这种方法指出窗体的处理函数，意味着所有用名为“testwin”的WNDCLASS构造出的窗体都拥有同样的处理函数，如果构造了多个窗体，如何让函数区分到底当前在处理哪个窗体的消息？DM7—2中，第1行窗体处理函数的第1个参数hwnd指出了被处理消息属于哪个窗体。只要在函数中用判断hwnd的不同，即可做出相应处理，这样一个窗体函数可被多个窗体共用，却又能各行其是。

第 种方法为某类型窗体指定相同处理函数，这种方法可为某个具体窗体对象指定处理函数。也就是说，可以用同一个WNDCLASS结构创建出的窗体拥有各自不同的处理函数。调用函数SetWindowLong()即可，SetWindowLong(hwnd, GWL_WNDPROC, (LONG) wndProc) 将消息处理函数指针wndProc设定给句柄hwnd指示的窗体。消息处理函数怎样处理消息？从DM7—2第1行可知，它有4个参数。第1个已解释，是接收消息的窗体的句柄。第2个是无符号整数，代表消息类型，如宏WM_PAINT就是重绘消息。从winuser.h中可知，该宏声明为#define WM_PAINT 0x000F，即说明十进制数15代表重绘消息。一个消息可能包含相关参数，则由第3、4个参数指出，其类型为指针，但很多时候被直接当成整数。比如，鼠标左键按下消息WM_LBUTTONDOWN中，wParam的整数值代表了此时其他键按下的状态，如MK_SHIFT代表键盘Shift键是否按下。读者可能会问，如果有2个甚至以上的参数需要表示怎么办？系统非常节约，比如还是WM_LBUTTONDOWN消息，需要包含此时鼠标的坐标x和y，它用lParam的高2字节代表y，低2字节代表X。如果还有更多信息呢？因为wParam和lParam都声明为指针，那么就可以将任何结构体的指针作为wParam或lParam传递即可。这与第5章线程函数的参数只接收一个void*类型却可传递任何信息是一样的。

《老码识途》

编辑推荐

《老"码"识途:从机器码到框架的系统观逆向修炼之路》包含不少工业级或非公开案例，读者不仅能以底层观和调试技巧解决各种实际问题；还可掌握一套学习方法，如“猜测—实证—构建”，调构学习法。

精彩短评

- 1、帮同学买的，他非常喜欢收集计算机的书
- 2、估计要到一定的经验和水平才能很好地消化或理解其精髓
- 3、本书十分适合于有一定功底的人看，从底到上，深入理解高级语言，顺便深入理解了计算机软硬件系统。使我对计算机的理解又深了一层
- 4、深入理解C语言，很好的书。
- 5、正好用得上，不错的书。
- 6、不错，看软件调试之前可以看看此书
- 7、正版买了5本平均下来34一本，当当便宜不解释啊
- 8、书内容很给力，就要看看书的人给不给力了。
- 9、我老师的书籍，早关注了。看了一段时间，终于还是买了。老师再接再厉。
对当当意见，以前买了个代码大全2，看了一年多终于看到后面，发现有一章少了一百多页。这么严重的问题就给我说超过一年了，不予售后服务。我于是郁闷好久，就不怎么在当当买书好久了。评论我也不喜欢写。不过还是发一下吧。别憋着。
- 10、相对偏底层，结合反汇编的书看效果好点
- 11、有作者自己的思想,有看头。
- 12、里面很多页字迹掉色了，好贱啊
- 13、适合了解底层细节。
- 14、粗略地看了一下，内容还行，但没有特别出彩的地方，没有让人眼前一亮的地方，反而有些地方像是在凑字数，赶时间。值不值得买，这个不好评价，见仁见智吧。
- 15、很用心，都是干货，强烈推荐
- 16、实践方法对程序运行的理解通透！
- 17、内容说不上很精彩，但比起那些抄来抄去的书还是有自己的特色。
- 18、我们《软件开发环境》老师写的书，先教你通过反汇编来分析、修改、自己写底层机器码，后面着重探讨面向对象特性在底层的实现和体现。

知识点都是底层的干货，对理解高层封装出来的一些概念的本质非常有帮助。比如指针本质上就是个4字节的地址，指针类型只是由编译器识别，然后体现在控制访问多少个字节的CPU指令上；比如函数是怎么实现调用、传参、返回的，传参又有寄存器传值、压栈传值、压栈传地址等方式，跨语言调用函数时调用惯例的协调。

总之弄懂了这些底层的机制，对高层语言的理解会透彻很多。

不过最好有一点汇编基础再读，否则略艰涩。

另一个特点是全书一直贯彻一种“猜测——实证”的思想，跟作者交流过这本书好几次，感觉这种思想是他最想传达的东西。

19、勘误表

http://blog.sina.com.cn/s/blog_7d5a09f90101dukr.html

- 20、大概翻了翻。逻辑性挺强的一本书。还没有细读。但愿不会失望。
- 21、这是一本非常需要带着批判思维阅读的书。
- 22、写的通俗易懂，截图非常好
- 23、很不错的书，可以看看
- 24、看了一部分，是偏windows的，内容的形式感觉就像是韩宏老师写的技术研究日志一般，呵呵，虽然工作有些年头了，但从中也能学习些东西。
- 25、成电韩老师的著作 顶起..... 成电 妓院 雄起.....
- 26、讲得很好，很细，值得一读
- 27、快递给力，当当给力，剩下的就看自己（看书）给力不给力了。

《老码识途》

- 28、内容跨度太大，不适合用来深度学习，拿来看看，作为了解和扩展思路比较好。
- 29、书看了一些才来评价的
- 30、该书的内容不错，学会了很多东西。可惜的是，书中有很多页印刷的时候叠在一起了，那部分根本看不成。开始没有发现，前面已经用笔画了，也不好换。建议大家书到以后大致翻看一下，如果有破损或印刷错误好及时更换。
- 31、现在看这个有些早了，得有基础
- 32、仔细看后应该会很有收益，目前就看了第一章，对于非软件非硬件专业人士来说还是有点难度。
- 33、韩老师叫我买的，简单看了一下，适合我这种喜欢折腾底层的人
- 34、很专业的书籍，老公买的
- 35、这本书真心很不错，我只看了前面，感觉很不错。
- 36、所谓万变不离其中，一通百通就是这么来的，研究的很深透！
- 37、对于了解编码底层知识非常有效。
- 38、从底倒上,从基础到系统,介绍了一种"猜测-构建-实证"的实践之路,很好,很适合想真正了解计算机的学生,专业人士看~~
- 39、内功方面的书
- 40、成为老马的捷径！
- 41、这本书我们老师写的，写的很好，推荐~~~//第一次在当当上购物，感觉很棒。东西很快就到了。且比亚马逊便宜一些。加油当当。
- 42、去年看过很好的一本书之一
- 43、好书，值得推荐.朋友推荐的。
- 44、一般吧 看起来有点乱 不习惯
- 45、很好的书，可惜还没看完就要还了，打底层基础必备
- 46、国内少见的程序员必读书籍。适合有一定基础的C程序员看，最好对汇编语言有一些了解。总之，这本书66的。
- 47、这本书是国内目前出的少有的好书。大概看完，作者非常注重实践，注重自我思考和推导能力的培养。底层实现介绍非常仔细，引导思考，真可谓手把手教了，学习起来很容易。读完这本书影响最深部分应该是作者介绍C语言一些语句的汇编码实现分析、C++基于C的面向对象思想由来，类的实现分析、虚函数分析，手把手的基本调试。感慨很多，在学校很少有老师这样讲，知其然，知其所以然。推荐正在学校学习编程的，或者正准备找工作的、或者刚从学校出来的，或者工作了仍然对这些认识不深的人，都来读读，绝不会让你失望。当然，这本书也有他的不好，就是排版不太好。希望后续版本能把排版好好弄弄。
- 48、程序员枯燥的生活就是要学习一些死的要命的东西，没法子，机器底层只认一对一翻译的东西，想偷懒机器不答应。
- 49、也没全看，刚买了，不好说好坏
- 50、看了章，对基础原理讲的比较透彻。
- 51、这本书挺好的，从机器码讲到上层的框架，计算机的同道们一定不能错过哦
- 52、书刚看一点，内容比较详细，真的不错，非常适合。。。。。
- 53、国内难得一见的好书，细致明白，作者应该做过教育
- 54、从底层汇编开始讲起，慢慢一点点看还是不错的
- 55、看了下 三星作为鼓励 新手可以看看 深度不够
- 56、C程序员的书
- 57、确实很好，作者是用心的。这本书适合修炼内功用，相信看后能对系统又一个更深的理解。书中说看这本书不需要汇编语言的知识，但显然如果能有汇编的知识会更好，特别是32位汇编。
- 58、前几页还行，后面看不懂，汇编太复杂了
- 59、从底到上，难得国内的好书，很实际。
- 60、很少能在国内见到这这高分量的书籍啦，看完本书，可以对整个软件环境有更深一层的认识，强烈推荐大家来看看

- 1、我们《软件开发环境》老师写的书，先教你通过反汇编来分析、修改、自己写底层机器码，后面着重探讨面向对象特性在底层的实现和体现。知识点都是底层的干货，对理解高层封装出来的一些概念的本质非常有帮助。比如指针本质上就是个4字节的地址，指针类型只是由编译器识别，然后体现在控制访问多少个字节的CPU指令上；比如函数是怎么实现调用、传参、返回的，传参又有寄存器传值、压栈传值、压栈传地址等方式，跨语言调用函数时调用惯例的协调。总之弄懂了这些底层的机制，对高层语言的理解会透彻很多。不过最好有一点汇编基础再读，否则略艰涩。另一个特点是全书一直贯彻一种“猜测——实证”的思想，跟作者交流过这本书好几次，感觉这种思想是他最想传达的东西。
- 2、这几年，有过几本类似的书，希望把c语言讲明白。但是，没有达到这本书的程度。这是一本含有真知灼见的计算机学习指南，可以把你从码农变成系统工程师。如果你不想成为一位码农，无论是刚开始接触计算机，还是已经工作了，如果对计算机一知半解，拿起这本书从头到尾好好读一遍。如果需要的话，可以再读两遍。你会看懂以前看不懂的内容，你会看到以前看不到的地方。掌握了这本书的精髓，你会真正了解计算机和计算机语言。谢谢书的作者。
- 3、看完第一章时的感觉是惊喜，但是看完整本书之后就只剩下还行了。前不久读着读着实在忍不住了就在饭否上吐槽到：『（作者）沉迷于用底层的方法去分析，但是又囿于各种细枝末节；到了高处又缺乏一定的核心抽象部分的分析』。书中很多问题看似用底层分析的方法解决得比较巧妙，但是这些分析方法依赖的基础完全就已经足够解决问题本身，这样一来就有种为了底层分析而底层分析的意图。另外这本书的定位也有点问题：从内容看，受众似乎是那些接触编程不久的学生，因此无论从作者强调的分析方法还是内容的编写逻辑上，个人感觉都对目标读者不友好。而对我个人而言，内容又显得深度不够。书中还有一些工程实践的观点/原则是错误的，代码说实在也写的挺烂的，真正想通过这本书提高码农能力的话恐怕是要失望了。最后，我非常推荐作者在书中用来解决问题的思路和方法（虽然仅对某一类问题比较高效），但是并不推荐内容本身。简言之，这是一本非常需要带着批判思维阅读的书。-----原评论-----意料之外。作者似乎是电子科大的老师，果然比起我的屌丝学院的老师高明不知道哪里去了。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu000.com