快速编程

高庆◎著

吉林大学出版社
JILIN UNIVERSITY PRESS

快速编程

高庆　著

吉林大学出版社
JILIN UNIVERSITY PRESS

图书在版编目(CIP)数据

快速编程/高庆著. —长春：吉林大学出版社，
2011.12
ISBN 978-7-5601-8081-6

Ⅰ.①快… Ⅱ.①高… Ⅲ.①程序设计
Ⅳ.①TP311.1

中国版本图书馆 CIP 数据核字(2012)第 000261 号

书　名：快速编程
作　者：高　庆　著

责任编辑、责任校对：陈颂琴　刘守秀　　封面设计：中尚图
吉林大学出版社出版、发行　　北京紫瑞利印刷有限公司　印刷
开本：880×1230 毫米　1/32　　2012 年 2 月　第 1 版
印张：5.75　　字数：120 千字　　2012 年 2 月　第 1 次印刷
ISBN 978-7-5601-8081-6　　定价：22.00 元

社址：长春市明德路 501 号　邮编：130021
发行部电话：0431-89580026
网址：http//www.jlup.com.cn
E-mail：jlup@mail.jlu.edu.cn

前 言

随着中国软件业的迅速发展，国内对程序员岗位的需求越来越大，要求也越来越高。软件质量的竞争已成为公司竞争的基础。软件质量与程序的设计、编码有很大的关系，所以各大软件公司一方面在不断提高技术人员的程序设计与编码水平，另一方面在人才市场上争夺高水平的技术人员。现在人才市场上不仅要求程序员能写程序，而且要能写出高质量的程序。本书从程序设计与编码的理论出发，将理论与实践结合起来，重点讲解实际工作中的思路与方法，使程序员不仅从实例中掌握到知识，更从实例中掌握到思路与方法。

本书在编写过程中，力图体现如下特色：

一是抓住重点。本书根据作者在工作中的经历，并亲自测试一些常用的方法与思路，找出编程工作中的重点，以避免掌握过多的对编程效果影响不大的技能而浪费时间。这样经过本书的阅读后，读者的编程水平会有立竿见影的效果。

二是尽量简洁。本书使用通俗的语言，将一些较好的思路、方法进行简明易懂的描述。这样有利于读者理解，使用相关的思路与方法，从而加快了学习效率。

三是实践性强。本书配有相关的实例，而这些实例的分析都是从具体的思考角度进行描述，力求更加真实地反映程序员编写时的具体实际状况，从而真实地表现了在实际中会碰到的问题及应该如何使用一些思路与方法解决问题。

阅读本书的建议：

1. 结合实际情况，阅读本书。因为本书的所有思路与方法与其特定的情况有关，不同的情况，其效果也不尽相同。所以读者要理解相关思路与方法的具体使用情景，并分析自己的具体情况，选择相应的思路与方法。

2. 在具体的编程过程中使用、体会书中介绍的思路与方法。实践是掌握思路方法等理论的好办法，所以要多进行编程实践，在编程实践中使用并体会这些思路与方法。

3. 多进行总结。在使用本书的相关思路与方法时，可能有的思路与方法用了后并不理想，这时可以总结其问题在哪里。另一方面，如果效果非常好，也可以进行总结，总结其合理的原因。这样就能加深对思路与方法的理解。

4. 了解各章的内容、重要性，及它们对编程的影响。

第一章“编码基本知识”，介绍了代码过程中的接

口、变量定义、流程、调试及程序员的状态。它是一个大体介绍，了解了这些就了解了编程工作的主要内容。

第二章“函数与API”，介绍了编程中用到的主要API类别，它说明了API的重要性。程序员要掌握的API不仅仅如此，这里给出了程序员总结API的思路，API掌握得好，写出代码就有了思路。

第三章“软件工程与架构设计”，介绍了编程工作的重要内容——明白何时进行编程工作，编程工作与其他工作的关系。软件工程与架构设计是编码之前必须弄明白的一些内容。不然，编程工作就容易失去方向。

第四章“代码思路”，具体介绍了编写代码时的两个重要的工具“接口”与“流程”，它们是进行编码的好助手。

第五章“常见警告与错误分析”，给出了几个实例，并进行了分析。读者可从中了解分析的思路。这样可以自己多积累一些警告与错误，这对编程效率的提高很有好处。

本书根据作者的实践经验，结合了业界人士的许多一手资料，并参考了很多相关书籍编写而成。在此向这些人士与书籍作者表示感谢。

由于作者才疏学浅，水平有限，书中疏漏、错误在所难免，敬请专家和读者批评指正。

高 庆

2011年7月

目 录

第一章　编码基本知识

1.1 程序整体

1.1.1 概述

程序整体指程序的整体布局。

在这里，“整体布局”的概念有多种。

首先，从面向过程的角度看，程序可由多个函数组成，程序的整体布局可以看做是程序主要函数的布局。

第二，从程序运行的角度看，程序可由多个处理流程组成。一个程序的流程根据不同的输入值而走不同的流程，程序的整体布局可以看做是程序不同的处理流程。

第三，从面向对象的角度看，一个程序的功能被分解到一个或者多个类中，程序的整体布局可以看做是多个类的组合。

如果从面向过程来看，程序整体可以由函数组成。函数是一种将多行代码封装成一个块，并将它取名，设置调用参数、返回值的一种结构。函数是程序整体的重要部分。掌握了函数，就理解了程序的功能。掌握了函数的组合，就理解了程序整体的结构。

如果从面向对象来看，程序整体可以由类组成，类

是实现方法与属性的抽象。它的方法是对属性的各种操作。与函数一样，类也是一种封装，但它是对属性与方法的封装，相对于函数而言，类更加安全。

如果从运行而言，程序整体是各种流程的组合，一个程序运行有很多种流程。各个不同的流程代表了一种运行方式。

1.1.2　程序整体的一些特性

程序整体具有一定的特征性和多样性。

程序整体的特征性是指程序在技术或者逻辑上的一些特点。从各个角度看，程序有不同的特点。比如从技术上看，每一个程序都是做获取数据、处理数据、存放数据的工作。而不同程序数据获取、处理、存放都会有所不同，并且有自己的特点。比如网络程序取数可能是通过其他的程序传送过来，处理数据可能是并发的方式，存放数据可能是传给其他程序。而从逻辑上看，每个程序都由多个小模块组成，而程序正是通过各个模块来完成业务功能的，不同的程序其模块的组合方式是不同的。

程序整体的多样性是指不同程序的程序整体是不一样的，即使同一种项目因为所采用的技术或者业务功能不同，也可能导致其存在着不同的方面，可能在各个方

面有很多不同的特征，所以导致了程序整体的多样性。最容易想到的就是很多计算机公司的软件改造升级，其中就可能改变了架构、技术、功能等。

程序整体的特征性与多样性反映了程序的个性与特性。不同的特征导致了多样性，它将各个程序相互区别开来，这就为程序员掌握程序提供了帮助。程序员可以了解所要掌握程序的特征与多种不同类型程序的特征，这样就能很好地知道当前学习的程序的类型与具体特征，这样就有条理了，也容易记忆与理解。

在实际的程序设计中，了解程序的特征可以从程序运行的软硬件环境、程序的模块组成、程序里的进程、线程、进程与线程上运行的模块、数据的获取、数据的传递、数据的处理、数据的存放等方面进行了解总结。掌握了这些，基本上就了解了程序的基本特征。

1.1.3 程序整体的流程

在上一节中，已经讲过，了解程序整体的特征时，可以从程序的进程、线程、程序的模块组合、及进程与线程上运行的模块等进行了解。为了了解程序的运行情况，软件里有一个专门的名词——流程。

流程是从人们工作中转移到程序设计中的，在软件工程中，有专门的流程图。在工作中，流程是一种步

骤，比如工作流程是工作的步骤；在程序中，流程也是一种步骤，只不过它不是人做事的步骤，而是软件在电脑中执行的步骤。

软件中的流程有三种类型，第一种是代码的流程，第二种是模块的流程，第三种是数据的流程。代码流程是一行一行代码的执行流程，软件工程中的流程图一般就是指代码流程。模块流程是软件人员从代码流程引申的一种程序理解方法，即在模块流程中，每一个流程不是代码行，而是一个模块，显然这个粒度比代码中的代码行要大得多。这样的好处是可以帮助项目人员从大的结构上了解程序，而不用太拘泥于细节。数据流程也是软件工程中的概念，它描述了程序的数据从各个系统、各个模块，甚至各个接口的经过的情况，它关注每个系统、模块、接口的数据输入与输出是怎样的。程序员从数据流程中可以看到程序的功能，因为程序的作用就是接收数据、处理数据、输出数据。

了解程序的流程有利于了解程序的特征、功能。

每个程序可能有很多的流程，因为程序的流程根据入口条件的不同，走不同的执行路径，每一个选择都可以产生两种流程，而N个选择就是$2N$个流程，可能人工都无法统计完，所以总结流程时，要通过选择符号来标出其两种情况，而不是列出一个完整流程。

流程具有实时性，即只有在程序执行时才能知道程

序的具体流程。虽然有时可以通过非常细致地了解数据及各种环境来较为准确地估计流程，但是毕竟与实际情况有所差别。比如软件的bug就是预先估计与实际不符的实例，所以流程并不太容易在运行之前完全估计。

1.1.4 程序整体的部署与执行

程序整体的部署是指程序运行时的硬件、软件环境。硬件环境指程序运行时所用到的硬件信息，比如计算机、计算机的内存信息、处理器、硬盘、显卡等。有的程序的硬件环境复杂，比如有的程序可能需要多台计算机、特殊的通信网络等。

程序整体的执行指程序运行时的进程、线程、进程与线程上执行的模块、数据的获取、数据的传递、数据的处理、数据的存放等信息。

程序整体的部署与执行可以帮助我们从物理实际的角度查看程序的特征，加深对程序的理解。

与架构和编程技巧相比，程序的部署与执行更接近实际，更容易理解与思考。所以对于程序员来说，掌握程序的部署与执行更加有实效性。

1.1.5　了解程序整体的重要性

程序整体是一个较抽象的东西，没有很好很细的办法来控制它，所以在真正实践的编程中，无法保证程序整体规范性及正确性。

程序整体的特性、流程、部署与执行，程序员要足够地重视。这一点可以从文档中看出，在概要设计文档中会列出所用到的函数、类，并列出流程图，这些设计文档为下面的详细设计与编码提供了坚实的依据。

程序整体对开发经理，开发人员，维护人员都有很大的帮助。

开发经理可以从程序整体中发现系统的优缺点，这样可以在早期就坚持或者改正其中的设计。

开发人员可以通过程序整体保持自己的详细设计的大体方向不会改变。

维护人员可以通过程序整体了解整个系统的组成，有利于快速了解细节，在发生问题时，有利于定位其问题。

另外，了解程序整体还有一些很实用的用途，比如，当程序员跟别人交流所做的软件时，或者在参加面试时，在这些场合，不可能说得太细，而且说细了别人也听不明白，这时，如果能介绍其程序整体，别人容易理解，而且也能从中评价系统做得如何。

在进行设计时，无论是概要设计还是详细设计，函数、类、流程的设计都起到了核心的作用。其实整个设计就是围绕这几项进行。把这几项设计好了，设计就完成了。

当系统完工后，在性能、质量上如果有问题，那么，函数、类、流程的设计也会起到很重要的作用，如果想提高性能、质量，那么改进函数、类、流程最关键。

程序整体的重要性在于它是程序员或者其他工程人员对程序的第一印象，如果程序整体设计得好，那么第一印象就深，以后就能记在心里，但是如果很乱的话，就容易忘。

另外，如果程序整体不好，就不容易深入下去。

如果程序员、架构师认识到了程序整体的重要性，那么在作架构程序时、写代码时，就应作出容易理解、清晰的结构，这会对将来的维护、改造、调试带来很多好处。

1.1.6 思路

程序整体的思路是由外到里，由粗到细。就一个程序来说，如果想理解一个程序的整体，首先要弄明白其部署与执行，再弄明白其流程，再弄明白其特性，然后再分别从粗到细将其部署与执行、流程、特性中的东

西弄明白。这些相关知识可参考前面的章节。比如程序整体的部署与执行要弄明白其软硬件信息，程序执行时的进程、线程，及进程与线程上执行的模块，数据的获取、传递、处理与存放等。

理解程序整体的思路可以不固定，但最好形成一定的习惯，即从外到里，从粗到细的思路，这样就不容易出现遗漏或者想当然，以保证程序整体的正确与较好理解。

思路的作用是大家对它有固定的看法，从而可以指导我们理解程序整体。

当写一个程序时，如果没有思路，则无法进行下去，如果弄明白了思路，下面就容易想了，照着思路来即可。

程序整体与部署、执行、流程、特性是互为帮助的，比如，理解流程就会对理解部署有帮助，反过来，理解好部署，对理解流程也有帮助。

而各个部分的理解又需要程序员细心地理解其相应的内容。这些在前面章节中有介绍。比如在理解流程时，要理清其各个代码行的执行路线，要理清模块的输入输出及执行路线，还要理解其数据的流向路线及处理转变情况。此时应使用一些软件工程的知识加强理解，比如查看流程图、数据流图等。

程序整体的例子非常多，一般表示非常容易，但是