希望
Beijing Hope Electronic Press
www.bhp.com.cn

游戏创造

C游戏编程从入门到精通（第2版）

MANAGED C GAME PROGRAMMING【2ND EDITION】

北京希望电子出版社 总策划
浦 滨 编 著

科学出版社
www.sciencep.com

希望
Beijing Hope Electronic Press
www.bhp.com.cn

游戏创造

C游戏编程从入门到精通（第2版）

MANAGED C GAME PROGRAMMING【2ND EDITION】

北京希望电子出版社 总策划
浦 滨 编 著

科学出版社
www.sciencep.com

内 容 简 介

本书讲述如何进行游戏编程的详细过程，系统地介绍了基于C/C++环境游戏设计与开发的方法和技巧，内容全面并相互关联、渗透。以基于不同平台的游戏制作为背景，知识阐述与具体案例相结合，深入浅出，直观、实用、可操作性强；是一本集入门、深入到精通C/C++游戏编程的综合性图书。

该书将大学计算机及大量相关专业课程的知识运用到游戏编程的实践中，并详细诠释计算机及其他专业课题，内容涉及计算机硬件、软件等方面的编程技术。具体内容包括：图形绘制、动画实现、中文显示、文件调用、内存使用、声卡控制、中断和内存驻留技术、接口技术、数据库实现、简单病毒、界面技术等，并且配以大量源程序以及程序注释，对所涉及理论进行充分的讲解和支撑。

本书根据大学C/C++语言教学需求，适合高校计算机和数学相关专业师生。对于C/C++语言的初、中级用户，业余爱好者学习与培训，以及有一定软件开发经验的程序员、专业技术人员也有很好的借鉴和参考价值。

随书光盘内容为书中实例的源代码及可执行程序。

图书在版编目（CIP）数据

C游戏编程——从入门到精通 / 浦滨编著. —北京：科学出版社，2006

ISBN 7-03-015751-6

Ⅰ.C/C++… Ⅱ.浦… Ⅲ.C/C++—程序设计

Ⅳ.TP321

中国版本图书馆CIP数据核字（2005）第026899号

责任编辑：但明天 / 责任校对：李 真

责任印刷：媛 明 / 封面设计：刘孝琼

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

北京市媛明印刷厂印刷

科学出版社发行 各地新华书店经销

*

2006年8月第 一 版 开本：787×1092 1/16

2006年8月第一次印刷 印张：27

印数：1—3 000 字数：629 368

定价：39.00元（配1张光盘）

前言

毋容置疑，游戏编程就是编程！如果你接触过程序设计，那么打算写一个游戏已经不是什么困难的事情。如果认为自己虽然学过C语言，而对编程只有一点初步的了解，根本谈不上编写游戏的问题。我们是否具有编游戏的基础呢？不过，相信在看这本书的时候，你一定具备了：

（1）编程的基础知识和一些基本方法；

（2）对编写游戏有强烈的愿望。

下面从游戏设计的基本流程，游戏软件开发、设计方案着手，探讨游戏编程的有关内容细节。当一款游戏的开发工作正式启动的时候，首先要解决的问题就是引擎。

引擎设计

引擎的开发往往由设计人员协助程序人员完成。这里要明确的是，游戏的好坏跟引擎有很大关系，引擎设计得是否合理，从某种程度上反映了游戏的水平。

游戏引擎是什么？一款游戏有漂亮的画面，逼真的音效，便捷的操作，还有普通用户不能看见的内存管理、设备管理等，这些事情由谁来完成呢？它们都是游戏引擎的工作。

引擎包含系统底层的程序、数据结构、调用方法等内容，这些直接限制了你以后的游戏好不好编。设计游戏引擎应从下面几个方面考虑。

1. 功能划分

任何一款游戏都有许多功能，比如攻击、使用物品、施放魔法、移动、键盘输入、更换装备等，这些都需要引擎来实现。

设计的时候，有必要考虑将功能进行分类和简化，并将某些功能的实现看成另外几个功能同时作用的结果。从最基本的功能着手设计，不断利用已完成的功能去实现新功能。

但不要将两个或多个功能混淆或相互依赖，它们之间应该是独立的，即每个在引擎里定义的功能都可以脱离其他功能而单独使用，其他功能的实现则只要调用这些功能的结果就可以了。

2. 物品清单

本来应该脱离引擎而存在，它可以通过脚本实现。这里所说的物品是构成世界的基本元素，也就是一些原始资源，比如男人、女人、动物1、动物2、植物1、静物1等。

有了这些原始资源，引擎就显得有意思了，并且可以用引擎构造世界。

3. 地图编辑器

把地图编辑器也包含在引擎中，其目的不是为满足玩家的需求，而是能够反复利用引擎去开发不同游戏。

既然我们已经有了一些原始资源，那么就拿它们来创造世界吧！如果觉得手头的资源太少，那么地图编辑器就派上用场了，可用以生成更多的资源，比如怪物、宝物、地形地图等。

4．留后门

游戏引擎应该提供一个后门为管理人员所用。不仅能提供一个无须编译而直接修改游戏内容的方法，而且为今后的测试提供了极大方便。

到此，设计工作已经差不多了——不要以为引擎的开发就这么简单，后续的大量工作还需程序员完成。这仅仅是一个向导或者说标准，制定这个标准的目的是为今后开发的方便，而不是跟程序员找别扭。

有了一个引擎，后面的工作变得轻松多了。

游戏规则

你的游戏之所以好玩，就在于游戏对每位玩家所采用的规则都相同。优秀的游戏必定有严谨的规则，无论你设计的游戏是什么，先得把最为主要的规则确定下来是关键。

1．胜负判定

不要认为胜负判定非常容易，其实只要游戏再复杂一点，胜负判定就会变得十分困难。比如当一颗炸弹同时炸死自己和仅剩的一个敌人的时候，或者当双方积分相同的时候，如何判定胜负？最简单的办法就是和局。即使如此，事先得有对胜、负或和局的判定。

2．公式算法

游戏中数值之间的换算关系更是需要好好研究，有时仅仅使用几个公式会将程序员引入歧途或尴尬的境地。

3．随机事件

游戏之所以吸引人玩的另一个原因就是随机事件。当随机事件发生的时候上帝都在祈祷，如何充分利用随机事件来让玩家体会更多的乐趣，的确需要好好考虑。当然，不同的游戏应该有更多的公式，这里无法一一列举出来。

剧情

游戏大部分是对现实生活的形象模拟，一些游戏有丰富的剧情，比如说 RPG。剧情设计往往是爱好者的看家本领，也是游戏耐玩的理由。

1．剧情长度

繁琐的、支离破碎的剧情会令玩家讨厌。如果无法保证剧情的质量，先考虑一下如何保证数量上的简洁，起码不会被骂做王二娘。

2．结局

很多玩家可能喜欢多结局的 RPG，有悲剧结局，也有喜剧结局，还有恶搞结局。结局处理上可以比故事情节多下工夫，无厘头的结局不失一个选择。

3．支线剧情

有的玩家不喜欢玩支线剧情，有的反而喜欢玩支线。设计时可以兼顾，主线剧情环环相扣，支线剧情费尽心思，两者穿插，这样可以满足两种玩家的要求。

法术、物品、属性及其他

玩家要得到最好的，我们就给他们最好的。作为交换条件，玩家要会付出更多的时间去泡点、去消费。

1．法术

法术不要弄得太多，要有针对性，不要将游戏做成 NWN 那样。每个人都可以从不同角度给 NWN 做出很高的评价，但真正窝在家里成天玩的不是 NWN，而是 TFT。

2．物品

究级装备、黄金宝剑、暗金套装、超级极品……你的游戏需要这些吗？为什么不呢？一切有利于赚钱的都值得考虑。

3．属性

星际和 TFT 是当今最火的游戏中耀眼的两个，值得称赞的地方太多了。有一点大家都很清楚，那就是属性的修改为每个版本必须做的工作。我们在设计游戏的时候也应着重考虑这个环节，它不仅可以让游戏趋于完美，更主要的是可获得很多免费的评论和宣传，也会招来很多新玩家。

4．其他

还有别的什么吗？需要提的太多了，怪物、Boss、迷宫……这些具体问题应根据具体游戏来确定。有一点是肯定的，那就是游戏要用来赚钱，而不是只给人玩死！

大家似乎已迫不及待想写自己的策划方案了吧？请不要太着急，还有一些非常重要的问题没解决。

界面与操作

一打开电脑就自动进入游戏吗？当然不要这样。用鼠标点击某个应用程序的图标，之后的一切就变得难以琢磨了。

1．界面

界面设计力图简洁、明了，能够让玩家一眼认出来哪里是 New Game，哪里可以 Save，哪里在 Load……

毫无疑问，还得在最为明显的地方放一个 Quit。不要在界面里面跟玩家玩捉迷藏。F1 一定要设计，但千万别在这里显露你的文笔，没有几个玩家愿意花十几分钟去欣赏你的 Help 杰作。

不能让玩家到处找按钮，直接用箭头给玩家指出来。有些按钮或状态栏隐藏在深层的菜单中，玩家一时半会能找到，一定要有演示动画指明方向。要知道，玩家停留在 Help 里的时候也是最容易放弃你的游戏的时候。

2．操作

采用通用的操作，比如鼠标左键选取，右键放弃。关闭按扭在窗口右上方或窗口底部明显的位置。

鼠标移动最好是左键走、右键跑。键盘操作最好是 W、S、A、D 或↑、↓、←、→。

设计师应尊重玩家的操作习惯，这样容易博得玩家的认同感。热键和自定义键位功能则是为高级玩家准备的，这些东西不必告诉新手，也没必要放在 Help 里，让玩家慢慢去摸索好了。

到这里一款游戏的设计工作已接近尾声，当然还有很多可设计的东西，接下来可以进入角色了，期待已久……

选择编程语言

游戏可以使用任何语言来编写，这是可以肯定的。

好的编程语言能够达到好的效果，游戏程序员的自信来自于其选中了一种方便、完美、高速的语言，但不局限于这种语言。

刷屏程序应该包含：背景屏幕刷新、精灵动画刷新、鼠标处理、键盘处理等，程序刷屏在 70 次/分以上，如果做不到，优化你的程序。

尽量少用乘除法、浮点数，必要时使用移位乘法。没有人会用数学描述去写一幅游戏图形，所有图形都来自美工画的 pcx、bmp，你只要负责如何做一个读写 pcx、bmp 的函数就可以了。

游戏开发是一个漫长的过程，没有一天能写成的游戏。

一般说来，我们做一个游戏用半年时间，其中两个月编引擎，两个月编游戏，剩下两个月调试，可见引擎的重要性。

游戏开发与实现

1. 游戏编程的要素

一个成功的游戏通常有如下价值的特点：

（1）极高的美工水平；

（2）逼真的动画效果；

（3）精巧的构思；

（4）简便的操作。

可以说这与编程的关系不算很大，只有部分动画和用户操作是需要在编程中下苦功夫的；而游戏成功的更重要原因是构思的独特和美工制作的赏心悦目。

以下是根据游戏的要素和编程的特点提出的游戏编程的五大要素：

- 动画效果　动画的效果一定要流畅，没有散动和尽量逼真，最好能够符合物理学和生物学的运动规律。
- 人机交互性　游戏易于操作，并且能够快速响应，必要的时候能够设计出适合游戏的输入设备和驱动程序。
- 媒体多样性　提高图像、动画、声音和操作的同步性、混合性。
- 数据结构高效性　用于实现游戏的数据结构一定要经过规划，尽量简便、高效，心脏适用面和扩充性强。
- 代码的通用性和对象化　引入面向对象的思路将对理清程序员思路、简化程序设计心脏程序扩充等问题带来相当大的帮助。

这些编程要素涉及直接写屏、中断、多任务、内存技术、动画技术、显示技术和优化

算法等编程技术；包含内存、PC 嗽叭、声卡、键盘、鼠标、手柄、显卡等各种硬件原理和硬件编程；涉及《C 语言程序设计》、《C++语言程序设计》、《汇编语言》、《数据结构》、《数据库》、《计算机硬件》、《接口技术》、《算法》、《高等数学》、《概率论》、《人工智能》、《软件工程》和《多媒体技术》等计算机专业课目；还涉及《动画技术》、《三维游戏编程》、《加密、解密技术》和《病毒与病毒防治编程》等一些流行的计算机专业技术。

具备良好的计算机专业知识是实现游戏编程的前提。不用太担心许多专业知识你还不太懂，这里是游戏编程，编程能力才是关键，相关知识都围绕游戏编程，之后的音节会根据相应需求有所阐述。

其实，游戏编程可以看作动画、输入输出设备和多媒体编程的集合，写程序前应当设计好数据结构及算法。

2．游戏的流程

别看游戏有那么复杂的情节和玩法，从开始到发展再到结束的流程中，归根结底就只有四样东西：

- 动画　根据变量运算结果对屏幕中变化的对象进行重画，从而实现动画效果；
- 响应　对于来自键盘、鼠标等输入设备和计算机内部如时间等中断进行响应；
- 运算　根据各类具体响应，通过计算和重新赋值来改变程序内变量数值；
- 循环　反复进行动画、响应和运算的操作来实现游戏的真正进程。

实现一个游戏的基本流程如图 A 所示。

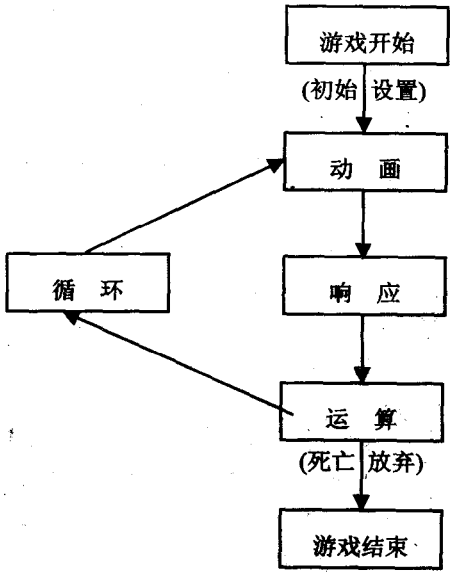

图 A　流程示意

下面以俄罗斯方块为例，动画流程分为以下几个步骤（图 B、图 C、图 D、图 E 及图 F）：

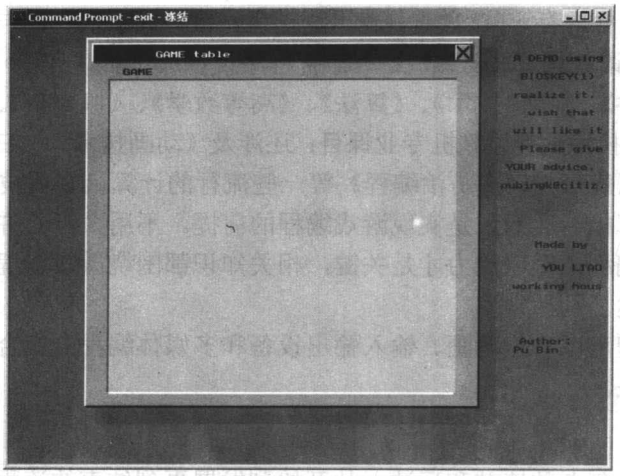

图 B　启动游戏

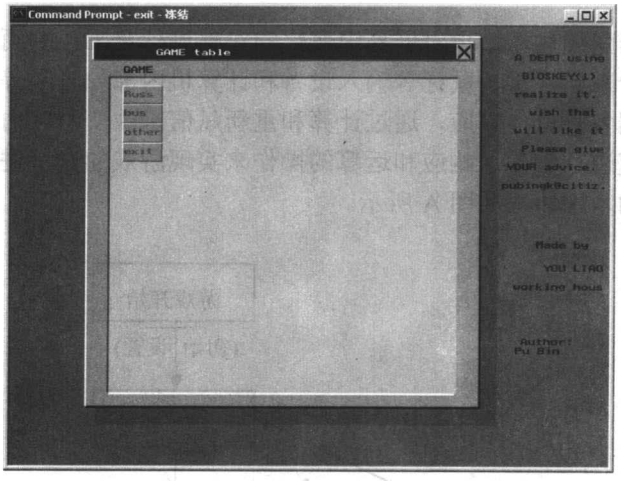

图 C　游戏初始化

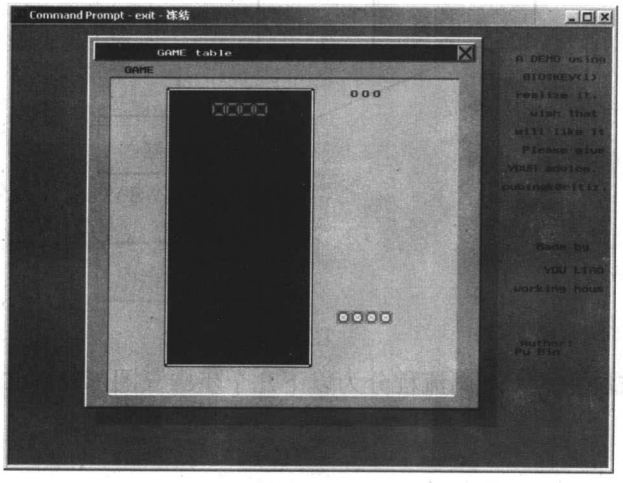

图 D　开始游戏

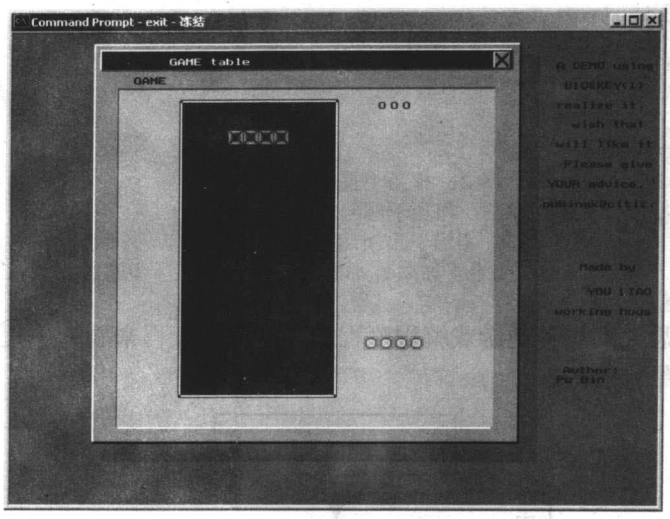

图 E　游戏进行中 1

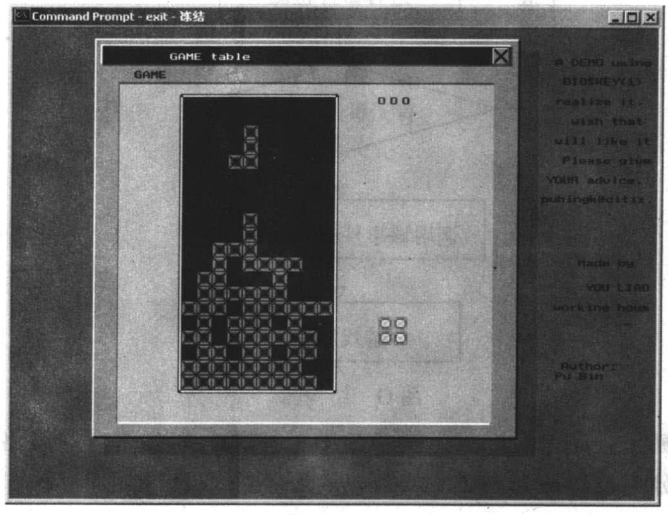

图 F　游戏进行中 2

（1）游戏开始，初始设置（这个过程设置了所有的变量初始值）；

（2）进行一个方块下落或者消去的动画（这个过程只负责重画屏幕）；

（3）然后察看是否有用户发出的旋转、下落、左右移动或退出游戏的指令（这个过程接收信息，同时存入输入变量）；

（4）此后如果有用户指令根据其指令计算出块的下一个形状、位置，同时也按照游戏本身设置的下落速度进行叠加计算并且判断是否到达底部，如果到达底部再判断是否可以消去和是否死亡（这个过程改变各种变量的值）；

（5）计算完毕后，根据结果进行下一次下落或消去的动画（回到步骤 2），如此循环往复直到游戏结束（这个过程就是一个 while 语句之类的循环）。

对应上面 5 个步骤，再以一个简单的 C 语言伪程序为例：

```
#include...
#define...
```

```
void main(void) {
  int a,b,c;              //步骤 1：相当于设置初始值
  a=1;
  b=2;
  c=3;
  while(a!='q') {
    printf("%d",c);  //步骤 2：相当于重画屏幕
    a=getch();       //步骤 3：相当于响应输入设备
    c=a*b;           //步骤 4：相当于重新运算变量值
  }                  //步骤 5：相当于如果 a 不等于 q 就继续循环步骤 2~4
}
```

面向对象编程思路的融入是游戏编程流程的又一重点。再来看一下程序设计，原先的流程可以被理解成结构图 G。

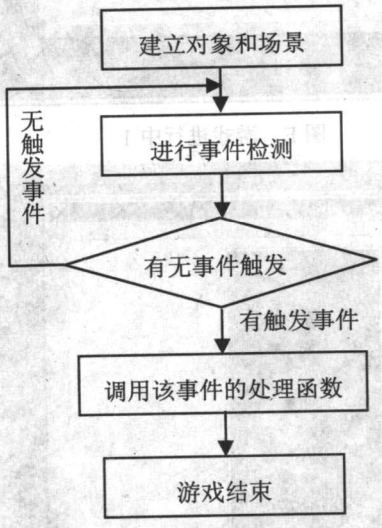

图 G　结构图

这和前面游戏流程并不矛盾，只是换个角度看问题。表 A 是对象思路中的每个环节和游戏流程的四大部分进行对应关系。

表 A　流程与对象

游戏流程	对象思路
动画	对象事件函数中的对象动画处理
响应	事件检测（也有的加上对象函数中的响应部分）
运算	对象事件函数中的运算
循环	事件检测循环

一个是流程明晰，一个是流行的对象化思路，那么我们到底应按照哪种思路来编写程序呢？

由于我们使用的是 TC 平台，所以本书的思路是，在流程化语言的基础上加入面向对象思路。这样的好处是能让流程化语言也拥有一定的面向对象思路容易扩充、维护的优点。

本节就是给了你一套写游戏最简单的模版。无论你要写怎样的游戏，都将它分成动画、响应、运算和循环四部分,外加一个初始设置。只要记住给你的那个 C 语言伪程序，套用之

后再一步一步改写，一个游戏就出来了。怎么样，解牛的顺序知道了吧！

3. 游戏的设计步骤

软件工程课告诉我们任何一个软件的实现都要经过以下几个步骤：

- 构思　你的创意，具体的内容规划；
- 设计软件结构　用简洁高效的数据结构实现你的创意；
- 编写程序　将你的数据结构用程序语言来表达出来；
- 调试、测试　对程序的系统可用性、模块功能进行测试，检验程序中错误和边缘问题，同时修改程序不符人意的部分；
- 完善、改进和推出　使游戏更加好玩、更便于操作。

游戏设计事实上也是软件的设计，然而它又有自身的一些特点。比如，游戏强调有极其新颖的创意，要对游戏的每个细节都有准确、详尽的描述和表现等。游戏编程的步骤可以归纳为如下几点。

（1）创意阶段

良好的创意。对创意进行完善，设计出游戏的完整过程并画出流程图，并将游戏分解为若干层次。

对每个层次进行详尽的描述，包括其中的人和物以及每个层次的实现目标。

对每个层次设定规则，包括人物的移动、操作者的权利、游戏的奖励和惩罚以及周边的一切环境、音效和帮助。

（2）规划阶段

使用的图形模式，比如 BIOS 中断 10H 的 13H 模式，即 200*320 像素 256 色。

确定图形、动画的复杂程度，比如用二维实现，还是三维。

考虑使用游戏函数库中的哪些函数来实现游戏创意中的那些要求。

（3）周边准备

制作人物、环境的图像文件，用绘画软件制作*.jpg 或*.bmp 文件。

设计人物、环境的声音文件，用录音机制作*.wav 文件。

建立地点、发展层次，人、物品、对话和奖励惩罚的数据库文件。例如用 FoxBase、dBase 或者 FoxPro 数据库软件建立*.dbf 文件。

（4）细部实现

图像实现函数、图像动画函数、声音播放函数、输入装置驱动和功能函数以及存盘、读盘函数。

（5）模块实现

数据结构系统、图像、动画系统、输入/输出系统、人工智能系统、游戏循环系统、用户界面系统以及声音系统。

（6）整体完善

美工、音效的提高；游戏动画、运算效率的提高；游戏功能和完整性的扩展；游戏的文件大小的缩减。

这只是一个大的方向性原则，具体的内部设计步骤根据不同游戏的情况和个人风格进行增减或调整。然而关键的是要编写一个好的游戏一定要按照软件工程的步骤，从构思到

结构到编程一步一个脚印；切不可贪快——一上来就写，到头来程序写到一半写不下去，前功尽弃。好的程序员才可能成为好的游戏编程人员。庖丁要顺骨解牛，不可乱来！

本书从 C/C++语言游戏编程入手，对图形绘制、中文显示、动画实现、文件调用、内存使用、声卡调用、中断、内存驻留技术、接口技术、数据库实现、简单病毒、界面技术等进行了详尽的介绍，并且配以大量的源程序以及程序分析对所涉及的理论进行充分的讲解和支撑。

该书将大学计算机及相关专业大量课程的知识运用到 C 语言游戏编程的实践中。这些课程包括：C 语言程序设计、C++语言程序设计、汇编语言、高等数学、计算机硬件、接口技术、数据库原理、数据结构、算法设计、软件工程。此外，还涉及到图形技术、动画技术、病毒与病毒防治编程等一系列专业知识。

本书共包括 17 章，具体内容分别为：

第 1 章 文本模式游戏，通过一个简单的游戏引导大家认识游戏编程。

第 2 章 图形图像绘制，介绍 C 语言自身提供的绘图函数。

第 3 章 简单动画实现，利用 C 语言绘图函数制作简单的动画效果。

第 4 章 简单图形游戏，在动画的基础上实现简单的图形游戏。

第 5 章 实现图形模式，介绍在制作游戏前如何将屏幕从文本模式设置为图形模式。

第 6 章 绘制二维图形，介绍如何自行设计基本图形函数。

第 7 章 中文输出，主要介绍在图形中如何实现中文。

第 8 章 载入图形图像，图形文件的调用和显示，介绍 bmp、pcx 和 ico 文件的算法和显示。

第 9 章 动画原理及实现，全面介绍各种动画实现的方法，并给出例程。

第 10 章 子画面技术，介绍最流行的子画面动画技术。

第 11 章 文件操作，文件的建立、写入和读取，主要使用于游戏进度保存和游戏数据读取。

第 12 章 声音技术，主要介绍扬声、声卡的使用，以及在游戏中的实现背景音乐。

第 13 章 内存技术，主要介绍 xms、EMS 内存技术及其在游戏中的应用。

第 14 章 接口技术，主要介绍键盘、鼠标和串口的驱动和使用方法。

第 15 章 界面技术，为游戏提供具有面向对象思路的图形、操作界面。

第 16 章 编程思路及其他问题，通过加密和病毒保证游戏的版权。

第 17 章 游戏设计实例，通过一个实际的游戏编写来整合前面各章节学到的内容。

从构建一个全方位的 C 语言游戏实用函数库，使得函数数量达到 200 个左右。函数类别包括：图形函数、动画函数、图形文件调用函数、数据文件处理函数、中文显示函数、内存函数、声音函数、计算机基本功能设置函数、接口函数、界面函数、数据库函数等。需要说明的是，本游戏函数库是在对 Andre LaMothe 等编写的运行于 Microsoft C 7.0 平台下的函数库进行平台改写、函数大量修改和扩充以及增加了 100 多个其他方面函数的基础上实现的。此外，本书还包括 108 个例程。

本书内容全面。游戏编程全方位阐述；涉及计算机各相关专业课程；适于编写各类型游戏，包括桌面智力游戏、视频战斗游戏、简单 RPG 游戏、简单联机游戏等。

在创新方面，将 TC 作为游戏编程平台。保护模式（TC 没有）等更适合写游戏，所以很少有人用 TC 写游戏；

从游戏角度学习 C 语言，从一个新的角度看待编程教学。教师通常从专家角度进行教学，于是在基础教学过程中往往和学生脱节（学生无法理解其很多表达含义）。本书的内容和程序有不少是从大一到大四过程中的笔记修改而来的，与读者学习进度非常接近，从而更容易吸收。

在过程性语言中更多加入面向对象思路。在游戏对象结构定义的时候融入更多属性、事件函数指针和链表指针，在子画面处理和界面对象处理中更多采用了事件驱动的思路，从而为今后转向 C++游戏编程铺平道路。

此外，将计算机相关专业知识通过 C 语言游戏编程的视角进行了解、学习或巩固。平时学习各科目内容较为独立或者仅仅基于理论上的联系，通过游戏编程我们可以将它们贯穿起来学习，触类旁通。

本书读者对象：大学计算机、数学及相关专业学生；游戏爱好者。

另外，本书适合大学 C 语言教学需要，内容衔接和难度上由浅入深，着重衔接 C 语言基础教材；也适合于高校计算机和数学相关专业的学生以及所有 C 语言爱好者。

目 录