

<<C++程序调试实用手册>>

图书基本信息

书名：<<C++程序调试实用手册>>

13位ISBN编号：9787505362147

10位ISBN编号：7505362143

出版时间：2000-10

出版时间：电子工业出版社

作者：(美)帕颇斯

译者：段来盛/等

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：<http://www.tushu007.com>

<<C++程序调试实用手册>>

内容概要

本书作为有关 Visual C++ Debugger 的专著，是一本非常难得的好书。书中深入地分析了开发不包含逻辑和语法错误的代码技巧以及调试程序的基本原理，介绍了开发和调试命令行代码的过程和方法，说明了关于定位、分析及修复编程错误的方法，介绍了开发 Visual C++ 程序时所遇到的特殊调试问题。

本书是所有软件工程师的必读书籍，也可作为大专院校师生的参考资料。

<<C++程序调试实用手册>>

书籍目录

第一部分 代码开发技巧

第1章 编写好的代码 .

1.1 谁需要本书？

1.2 教学方法

1.3 从何处开始阅读？

1.4 警告！

并非所有的C / C + + 编译器都完全相同

1.5 语言无关的设计工具101

1.5.1 准备

1.5.2 模型！

1.5.3 结构图、伪代码及IPO框图

1.6 好的程序设计的五点要素

1.7 规则意味着可以打破

1.7.1 安塞尔亚当斯（Ansel Adams）或毕加索（Picasso）

1.7.2 注释块

1.7.3 标识符：identifiers、IDENTIFIERS和Identifiers

1.7.4 间隔与缩进

1.8 数据类型

1.9 匈牙利表示法

1.9.1 MFC、句柄、控件及结构的命名规范

1.9.2 一般前缀命名规范

1.9.3 变量命名规范

1.9.4 应用程序符号命名规范

1.9.5 Microsoft MFC宏命名规范

1.9.6 库标识符命名规范

1.9.7 静态库版本命名规范

1.9.8 动态连接库命名规范

1.9.9 windows . h命名规范

1.10 操作符优先级

1.11 小结

第2章 使用编译器优化

2.1 编码的责任与编译器的优化

2.2 Microsoft Visual C + + 的优化

2.2.1 调度指令

2.2.2 函数级连接

2.2.3 字符串池

2.2.4 使用register键字

2.2.5 常量和复制的传播

2.2.6 消除死代码和死存储

2.2.7 删除冗余于表达式

2.2.8 优化循环

2.2.9 降低强度

<<C++程序调试实用手册>>

- 2.2.10 inline键字的使用
- 2.2.11 省略帧指针
- 2.2.12 关闭堆栈检查
- 2.2.13 覆盖堆栈
- 2.2.14 函数调用之间允许使用别名
- 2.2.15 全局优化
- 2.2.16 产生内部函数的内联
- 2.2.17 优化math.h
- 2.3 Microsoft C++的优化开关
- 2.4 使用 Microsoft Visual Studio设置编译器选项
 - 2.4.1 Project Settings对话框中的 General类型
 - 2.4.2 Project Settings对话框中的 Code Generation类型
 - 2.4.3 选择结构对齐方式
 - 2.4.4 Project Settings对话框中的 Customize类型
 - 2.4.5 Project Settings对话框中的 Optimizations类型
- 2.5 建立发行版本的建议
- 2.6 小结
- 第3章 逻辑与语法错误
 - 3.1 好的调试策略
 - 3.2 四种程序错误类型
 - 3.2.1 语法错误
 - 3.2.2 连接错误
 - 3.2.3 运行错误
 - 3.2.4 逻辑错误
 - 3.3 查看错误消息
 - 3.4 预防性维护
 - 3.4.1 桌面检查的含义
 - 3.5 异常处理设计
 - 3.6 “请多多支持”
 - 3.7 Microsoft Visual C++的帮助
 - 3.8 小结
- 第4章 debugger
 - 4.1 确认Debugger可以使用
 - 4.2 启动Debugger
 - 4.2.1 Step Into和Step Over的区别
 - 4.2.2 Go
 - 4.2.3 Run to Cursor
 - 4.3 理解Debugger工具栏图标
 - 4.3.1 Restart
 - 4.3.2 Stop Debugging
 - 4.3.3 Break Execution
 - 4.3.4 Apply Code Changes、Edit and Continue
 - 4.3.5 Show Next Statement
 - 4.3.6 Step Into
 - 4.3.7 Step Over
 - 4.3.8 Step Out
 - 4.3.9 Run to Cursor

<<C++程序调试实用手册>>

- 4 . 3 . 10 Quick watch
- 4 . 3 . 11 Watch
- 4 . 3 . 12 Variables
- 4 . 3 . 13 Regisers
- 4 . 3 . 14 Meomry
- 4 . 3 . 15 Call Stack
- 4 . 3 . 16 Disassembly
- 4 . 3 . 17 Debugger Toolbar Menu Equivalentents
- 4 . 4 其他Debug菜单选项
 - 4 . 4 . 1 Step Into Specific Function
 - 4 . 4 . 2 Excmpions
 - 4 . 4 . 3 Threads
 - 4 . 4 . 4 Modules
- 4 . 5 本地菜单Debugger选项
 - 4 . 5 . 1 List Members
 - 4 . 5 . 2 Type Info
 - 4 . 5 . 3 Parameter Information
 - 4 . 5 . 4 Complete Word
 - 4 . 5 . 5 Go Definition / Refefence
 - 4 . 5 . 6 Go To Disassembly
 - 4 . 5 . 7 Insert / Remove Breakpoint.
- 4 . 6 Debugger窗口
 - 4 . 6 . 1 Trae窗口
 - 4 . 6 . 2 Watch窗口
- 4 . 7 View菜单和Debugger窗口
 - 4 . 7 . 1 Workspace
 - 4 . 7 . 2 Output
- 4 . 8 以不同的数据类型查看观察变量
- 4 . 9 打开Just - in - Time调式
- 4 . 10 Options窗口中的Debug标签
 - 4 . 10 . 1 Hexadecimal Display
 - 4 . 10 . 2 Source Annotation
 - 4 . 10 . 3 Code Bytes
 - 4 . 10 . 4 Symbols
 - 4 . 10 . 5 Parameter Values
 - 4 . 10 . 6 Parameter Types
 - 4 . 10 . 7 Return Value
 - 4 . 10 . 8 Load COEF & Exports
 - 4 . 10 . 9 Address
 - 4 . 10 . 10 Format
 - 4 . 10 . 11 Re - evaluate Expression
 - 4 . 10 . 12 Show Data Bytes
 - 4 . 10 . 13 Fixed Width
- 4 . 10 . 14 Display Unicode Strings
- 4 . 10 . 15 View Floating Point Registers

<<C++程序调试实用手册>>

- 4 . 10 . 16 Just - in - Time Debugging
- 4 . 10 . 17 OLE RPC Debugging
- 4 . 10 . 18 Debug Commands Invoke Edit and Continue
- 4 . 11 键盘映射
- 4 . 12 Debugger快捷键
- 4 . 13 小结
- 第5章 调试版本与发行版本
- 5 . 1 缺省的调试版本建立与发行版本建立设置
- 5 . 2 为调试版本建立修改工程设置
 - 5 . 2 . 1 修改调试选项
 - 5 . 2 . 2 修改产生调试信息的格式
 - 5 . 2 . 3 产生一个映射文件
 - 5 . 2 . 4 重定向调试输入和输出
- 5 . 3 什么是 . pdb文件？
- 5 . 4 什么是 . dbg文件
- 5 . 5 调试优化的代码
- 5 . 6 打开Debugger的另一种方法
- 5 . 7 使用基本版或调试版本
- 5 . 8 C / C + + 运行调试库
 - 5 . 8 . 1 旧版iostream . h和新版iostrearn之间的混乱
- 5 . 9 连接器参考资料
- 5 . 10 在调试版本中检测发行版本错误
 - 5 . 10 . 1 局部变量的自动初始化
 - 5 . 10 . 2 检查函数指针调用堆栈的合法性
 - 5 . 10 . 3 检查调用堆栈的合法性
- 5 . 11 TRACEN
- 5 . 12 VEAIFY宏
- 5 . 13 移植Visual C + + 旧的32位版本
 - 5 . 13 . 1 转换早期的32位工作空间和工程
 - 5 . 13 . 2 与Visual C + + 以前的版本共存
- 5 . 14 小结
- 第二部分 面向过程的环境
- 第6章 定位、分析和修复命令行代码错误
- 6 . 1 快速启动调试
 - 6 . 1 . 1 启动Debugger的快速方法
 - 6 . 1 . 2 变量初始化跟踪
 - 6 . 1 . 3 小心调试代码
 - 6 . 1 . 4 快速查看变量的内容
 - 6 . 1 . 5 中途停止Debugger
 - 6 . 1 . 6 执行到代码的指定行
 - 6 . 1 . 7 全速执行到一个断点
 - 6 . 1 . 8 运行至光标处
 - 6 . 1 . 9 现在测试
- 6 . 2 高级Debugger技巧
 - 6 . 2 . 1 使用新值运行
 - 6 . 2 . 2 循环调试技巧

<<C++程序调试实用手册>>

- 6.2.3 调用调试函数
- 6.2.4 递归调用与调用堆栈
- 6.2.5 查看反汇编代码
- 6.3 进一步观察变量
 - 6.3.1 使用QuickWatch窗口
 - 6.3.2 使用Watch窗口
- 6.4 小结
- 第7章 调试内联汇编语言代码
 - 7.1 汇编语言初步
 - 7.1.1 数据类型
 - 7.1.2 寄存器
 - 7.1.3 寻址模式
 - 7.1.4 指针
 - 7.1.5 协处理器
 - 7.2 调试
 - 7.2.1 减法运算
 - 7.2.2 使用256位整数
 - 7.2.3 程序循环
 - 7.2.4 使用协处理器求和实数
 - 7.2.5 使用协处理器计算正切值
 - 7.3 小结
- 第8章 在Windows代码中定位、分析和修复错误
 - 8.1 使用两台计算机调试
 - 8.1.1 准备远程目标计算机
 - 8.1.2 准备主计算机
 - 8.1.3 启动调试会话
 - 8.2 简明Windows入门
 - 8.2.1 基本的Windows代码
 - 8.2.2 调试文件详述
 - 8.2.3 程序执行的情况
 - 8.3 调试
 - 8.3.1 一个动画位图程序
 - 8.3.2 使用鼠标绘画
 - 8.4 小结
- 第三部分 面向对象过程的环境
- 第9章 定位、分析和修复命令行中的错误
 - 9.1 高级调试工具
 - 9.1.1 内存卸出
 - 9.1.2 定位错误参数从何处传递而来
 - 9.1.3 查找何处修改了指针
 - 9.2 Class View窗口要素
 - 9.2.1 ClassView窗口的 Grouped by Access功能
 - 9.2.2 ClassView窗口的 Base Classes功能
 - 9.2.3 ClassView窗口的 References功能
 - 9.2.4 ClassView窗口的 Derived Classes功能
 - 9.2.5 ClassView窗口中菜单的其余项
 - 9.2.6 ClassView窗口的 Properties功能

<<C++程序调试实用手册>>

- 9.2.7 在ClassView窗口中添加文件夹
- 9.2.8 在文件夹之间移动类
- 9.2.9 隐藏或显不 ClassView窗口
- 9.3 调试argc和argv []
- 9.4 小结
- 第10章 使用MFC类库开发Windows程序
- 10.1 为什么使用类库
- 10.2 一个真正的基础类——CObject
- 10.3 什么是应用程序向导和类向导
- 10.4 一个图形程序
- 10.4.1 使用AppWizard
- 10.4.2 使用 ClassWizard
- 10.4.3 建立AppWizard代码
- 10.4.4 AppWizard模板代码
- 10.4.5 在客户区的图形对象
- 10.5 剖面法
- 10.6 小结
- 第11章 定位、分析和修复 MFC Windows代码中的错误
- 11.1 内存问题
- 11.1.1 有问题的代码
- 11.1.2 定位和分析
- 11.1.3 修复工程
- 11.2 绘图问题
- 11.2.1 有问题的代码
- 11.2.2 定位和分析
- 11.2.3 修改工程
- 11.3 小结
- 第四部分 标准模板库 (STL)
- 第12章 STL编程实践
- 12.1 多体系结构 . .
- 12.2 掌握 C + +
- 12.3 STL 进退维谷的数据结构
- 12.4 初识 STL
- 12.5 STL和HP公司
- 12.6 大众化的 STL
- 12.7 STL总览
- 12.8 ANSI / ISO C + + 接受STL的过程
- 12.9 STL基本组件
- 12.9.1 什么是容器？
- 12.9.2 什么是适配器？
- 12.9.3 什么是算法？
- 12.9.4 什么是迭代器？
- 12.9.5 其他的STL组件

<<C++程序调试实用手册>>

- 12.10 完整的STL程序包
- 12.11 杂乱的C/C++家族
- 12.12 回顾数据结构
 - 12.12.1 静态与动态
 - 12.12.2 类型指针
 - 12.12.3 void指针
- 12.13 复习匈牙利命名法
- 12.14 函数重载
- 12.15 函数指针
- 12.16 运算符重载
 - 12.16.1 运算符和函数调用的重载
 - 12.16.2 编写自己的重载运算符
- 12.17 从结构到模板
 - 12.17.1 template关键字
 - 12.17.2 模板语法
 - 12.17.3 模板函数
 - 12.17.4 模板类
- 12.18 为什么STL比模板好
- 12.19 小结
- 第13章 定位、分析和修复STL代码中的错误
 - 13.1 从标准C++转向STL语法的过程中出现的问题
 - 13.1.1 用迭代器遍历容器
 - 13.1.2 仔细研究迭代器
 - 13.1.3 流迭代器
 - 13.1.4 为什么使用end()
 - 13.1.5 复制列表
 - 13.1.6 列表中的列表
 - 13.1.7 STL字符串指针的麻烦
 - 13.1.8 释放STL指针
 - 13.2 一个C++程序转变为STL语法的例子
 - 13.2.1 第一步 更新aSingleCard类
 - 13.2.2 第二步 更新WarDeck类
 - 13.2.3 第三步 修复STL代码的执行错误
 - 13.2.4 第四步 更新Opponent类
 - 13.2.5 第五步 运转的STL程序
 - 13.3 STL语法的源文件Wargame.cpp
 - 13.4 小结
- 第五部分 特殊的调试问题
- 第14章 使用DLL工作
 - 14.1 创建一个基于MFC的动态链接库
 - 14.1.1 头文件Framer.h
 - 14.1.2 源代码文件Framer.cpp
 - 14.1.3 建立Framer.dll
 - 14.2 创建使用DLL的主应用程序
 - 14.2.1 头文件DLLDemoView.h
 - 14.2.2 源代码文件DLLDemoView.cpp
 - 14.3 更加仔细地查看

<<C++程序调试实用手册>>

- 14.3.1 远程调试
- 14.3.2 有问题的代码
- 14.3.3 改正后的代码
- 14.4 小结
- 第15章 使用ActiveX控件工作
- 15.1 开发一个 ActiveX控件
- 15.1.1 使用Colltrolwizard
- 15.1.2 Test Container
- 15.1.3 产生一个真实的Clock控件
- 15.2 调试 Clock控件
- 15.2.1 准备远程目标计算机
- 15.2.2 准备主计算机
- 15.2.3 开始调试过程
- 15.2.4 查找问题
- 15.3 小结
- 第16章 调试 COM、ATL和DHTML
- 16.1 COM对象模型
- 16.2 创建一个ATL多边形工程
- 16.2.1 优化模块代码
- 16.2.2 测试控件
- 16.3 调试ATLCOM控件
- 16.4 小结
- 第17章 STL和 MFC编程
- 17.1 产生一个STL和MFC应用程序
- 17.1.1 复数
- 17.1.2 模板语法
- 17.1.3 基本的应用程序代码
- 17.2 调试
- 17.3 小结

<<C++程序调试实用手册>>

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:<http://www.tushu007.com>