# SoliDire文件打包器

# 详细设计说明书

姓名：李劼

班级：07409

学号：071202

邮箱：jokerleee@gmail.com

**纲要**

[SoliDire文件打包器 1](#_Toc230790391)

[详细设计说明书 1](#_Toc230790392)

[1 引言 3](#_Toc230790393)

[1.1 编写目的 3](#_Toc230790394)

[1.2 背景 3](#_Toc230790395)

[1.3 定义 3](#_Toc230790396)

[1.4 参考资料 3](#_Toc230790397)

[1.4.1 书籍 3](#_Toc230790398)

[1.4.2 资料 3](#_Toc230790399)

[1.4.3 源代码 3](#_Toc230790400)

[2 程序系统的结构（UML图） 4](#_Toc230790401)

[3 程序设计说明 5](#_Toc230790402)

[3.1 类设计表 5](#_Toc230790403)

[3.2 类关系图 7](#_Toc230790404)

[3.3 操作流程图 9](#_Toc230790405)

[3.3.1 打开压缩文件 9](#_Toc230790406)

[3.3.2 添加文件或文件夹 9](#_Toc230790407)

[3.3.3 打包所有文件或新添加的文件 9](#_Toc230790408)

[3.4 主要算法（伪码） 10](#_Toc230790409)

[3.4.1 遍历文件建立二叉文件树（递归） 10](#_Toc230790410)

[3.4.2 序列化文件树（递归） 10](#_Toc230790411)

[3.4.3 从序列化的文件树中恢复二叉文件树（递归） 10](#_Toc230790412)

[3.4.4 解压文件（递归） 11](#_Toc230790413)

[3.4.5 读写文件 11](#_Toc230790414)

[3.4.6 选择文件夹对话框（BROWSEINFO） 11](#_Toc230790415)

[3.4.7 显示文件树中的文件夹到ListVew（递归） 11](#_Toc230790416)

[3.5 压缩文件结构及打包/解包步骤 12](#_Toc230790417)

[3.5.1 压缩文件结构 12](#_Toc230790418)

[3.5.2 打包步骤 12](#_Toc230790419)

[3.5.3 解包步骤 12](#_Toc230790420)

[3.5.4 尚未解决的问题 13](#_Toc230790421)

## 引言

### 编写目的

* 复习及实践数据结构课程所学知识，加深对知识的理解。
* 练习C++语言的使用和面向对象的程序设计思想。
* 学习WindowsAPI的使用和用MFC设计windows程序
* 学习用UML语言的类图表示类层次结构
* 了解软件工程的相关知识。

### 背景

* 项目名称：SoliDire文件打包器
* 项目提出人：漆涛
* 项目开发者：李劼
* 项目目标：设计一个文件打包软件，可以将多个文件打包，并保留原有目录结构。

### 定义

列出本文件中用到的专门术语的定义和缩写词的原词组。

### 参考资料

#### 书籍

* 《深入浅出MFC第二版》 侯捷 著 华中科技出版社
* 《Windows API大全》 机械工业出版社
* 《C++ Primer Plus 5th Edition》 Stephen Prata著 人民邮电出版社

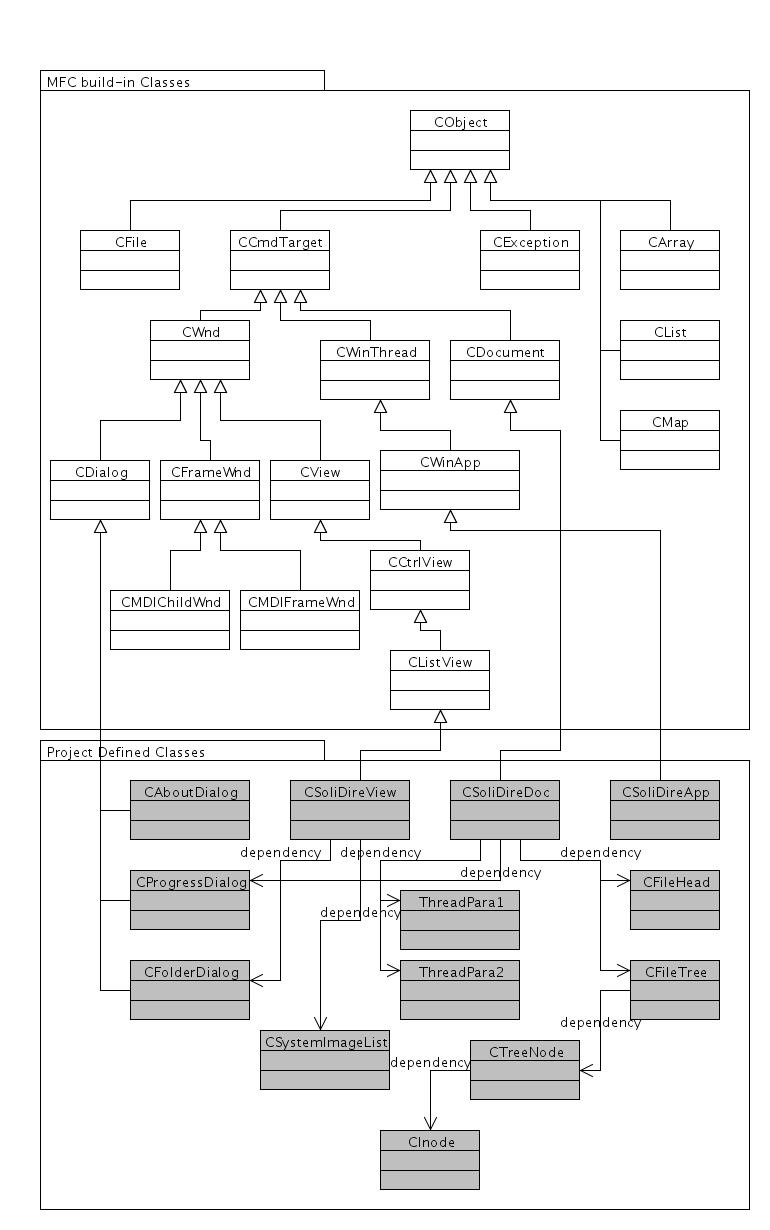
#### 资料

* 《MFC类库详解》 Microsoft Press
* 《STL Programmer's Guide》 www. sgi.com/tech/stl/
* 《范式哈弗曼编码的实现》 http: : blog.csdn.net/allenair/archive/2009/04/21/4097384.aspx
* 《SoliDire文件打包器概要设计说明书》 李劼 2009/4

#### 源代码

* “CFolderDialog”类源代码 by Mihai Filimon From www.codeguru.com
* “CSystemImageList”类 源代码 by Mark Otway From www.codeguru.com
* MyExplorer项目源代码 From www.pudn.com

## 程序系统的结构（UML图）

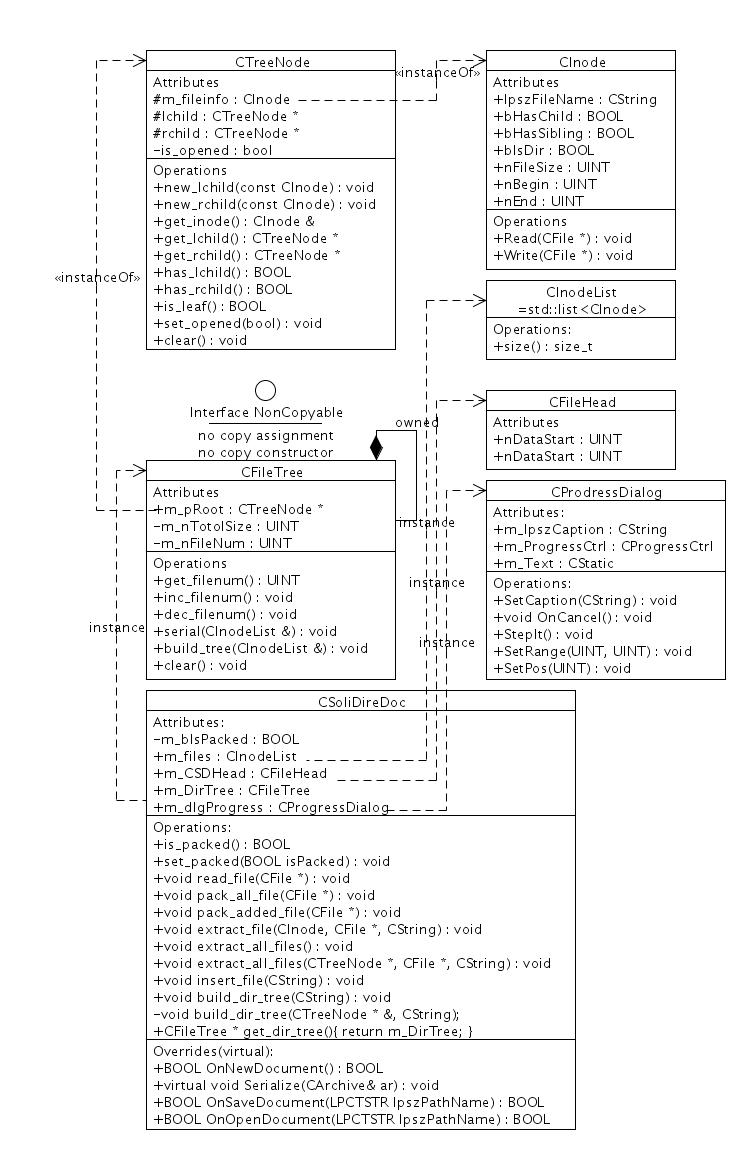


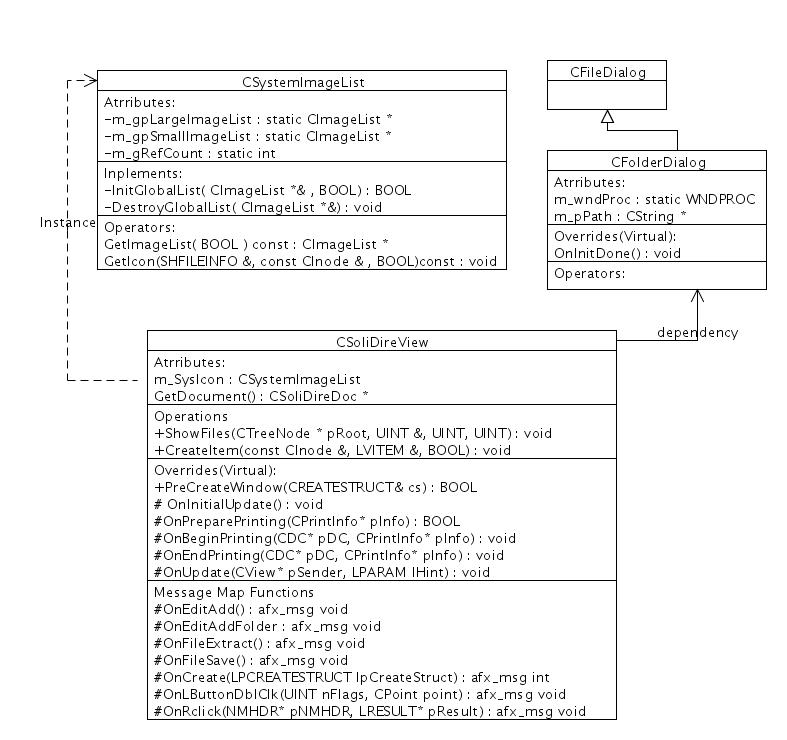
## 程序设计说明

### 类设计表

|  |  |  |  |
| --- | --- | --- | --- |
| 类名 | 依赖 | 成员及方法 | 功能 |
| **CMainFrame** | * CSysImageList | * CStatusBar m\_wndStatusBar : 状态条 * CToolBar m\_wndToolBar : 工具栏 * CImageList m\_ToolbarImages   : 系统图标列表 | 主窗口 |
| **CSoliDireApp** | NONE | NONE | 初始化应用程序 |
| **CSoliDireDoc** | * CFileTree * CTreeNode * CInode * CFolderDialog | * CProgressDialog ProgressDlg * CInodeList m\_files : 序列化的文件树 * CFileHead m\_CSDHead : 文件头 * CFileTree \* m\_DirTree : 二叉文件树   **public:**   * BOOL is\_packed() * set\_packed(BOOL isPacked):设置打包标记 * read\_file(CFile \*) : 从文件中读取树结构 * pack\_all\_file(CFile \*) : 打包整棵文件树 * pack\_added\_file(CFile \*): 打包加入的文件 * extract\_all\_files() : 解包所有文件 * insert\_file(CString) : 插入文件 * build\_dir\_tree(CString) : 从系统读取文件夹结构   **private:**   * void build\_dir\_tree(CTreeNode\*&, CString) * extract\_all\_files(CTreeNode\*, CFile\*, CString) * extract\_file(CInode, CFile \*, CString) * void pack\_file(CFile \*, CInodeList) | 处理数据：打包文件、解包文件、添加文件。 |
| **CInode** | NONE | * CString lpszFileName : 文件名 * BOOL bHasChild : 是否有孩子 * BOOL bHasSibling : 是否有兄弟 * BOOL bIsDir : 是否为文件夹 * UINT nFileSize : 文件大小 * UINT nBegin : 文件起始偏移量 | 存储文件信息的结构 |
| **CTreeNode** |  | * CInode m\_fileinfo : 文件信息 * CTreeNode \* lchild : 左孩子指针 * CTreeNode \* rchild : 右兄弟指针 * new\_lchild(CInode inode) : 新建左孩子 * new\_rchild(CInode inode) : 新建右孩子 * set\_inode(CInode inode) : 设置Inode * CInode & get\_inode() : 得到Inode结点 * CTreeNode \* get\_lchild() : 左孩子指针 * CTreeNode \* get\_rchild() : 右孩子指针 * BOOL has\_rchild() : 判断是否有右孩子 * BOOL has\_lchild() : 判断是否有左孩子 * BOOL is\_leaf() : 判断是否是叶子 * void free() : 释放文件树 | 文件树结点的封装 |
| **CFileTree** | * CInode * CTreeNode | * CTreeNode \* m\_pRoot : 文件树树根 * UINT m\_nTotolSize : 文件总大小 * UINT m\_nFileNum : 文件总数目   **private:**   * serial(CTreeNode \*, CInodeList &) : 文件树序列化 * build\_tree(CTreeNode\*&, CInodeList &) : 建立文件树   **public:**   * void serial(CInodeList &) : * void build\_tree(CInodeList &) * UINT get\_filenum() : 得到文件数目 * void inc\_filenum() : 文件数目+1 * void dec\_filenum() : 文件数目-1 | 二叉文件树，存储文件夹结构 |
| **CFolderDialog** | * CDialog * CFileDialog | * static WNDPROC m\_wndProc; * virtual void OnInitDone( ); * CString \* m\_pPath; * CFolderDialog(CString\* pPath); | 打开文件夹对话框 |
| **CProgressDialog** | * CDialog * CProgressCtrl | * void SetCaption(CString); * void OnCancel(); | 进度条对话框 |
| **CSoliDireView** | * CSoliDireDoc | * afx\_msg void OnEditAdd() * afx\_msg void OnEditAddFolder() * afx\_msg void OnFileExtract() * afx\_msg void OnFileSave() * afx\_msg int OnCreate(LPCREATESTRUCT lpCreateStruct) * afx\_msg void OnLButtonDblClk(UINT nFlags, CPoint point) * afx\_msg void OnRclick(NMHDR\* pNMHDR, LRESULT\* pResult) | 图形界面的显示和消息的响应 |
| **CSysImageList** | NONE | * GetIcon(SHFILEINFO &, const CInode & , BOOL)const : 得到某个文件类型的图标 * static int m\_gRefCount : 总引用数目 | 获得系统图标列表 |

### 类关系图





### 操作流程图

#### 打开压缩文件

#### 添加文件或文件夹

#### 打包所有文件或新添加的文件

### 主要算法（伪码）

#### 遍历文件建立二叉文件树（递归）

|  |
| --- |
| **BUILD-DIR-TREE(path, root)**  CFileFind fnd  bWorking = fnd.FindFile(path+”\\\*.\*”)  **while** (bWorking)  bWorking = fnd.FindNextFile()  root->new\_rchild(CInode(fnd.GetFilePath(), 0, fnd.GetLength()))  root = root->get\_rchild()  **if** (fnd.IsDirectory()) //如果找到子文件夹,递归处理  **BUILD-DIR-TREE**(root, path+fnd.GetFileName())  **else**  root->set\_inode(tmp) |

#### 序列化文件树（递归）

主要算法为树的前序遍历

|  |
| --- |
| **TREE-SERIAL(root, files)**  files.push\_back(p->get\_inode())  CInode & tmp = files.back()  **if** (p->has\_lchild()) //递归左子树  tmp.bHasChild = TRUE  **TREE-SERIAL**(p->get\_lchild(), files)  **else**  tmp.bHasChild = FALSE  **if** (p->has\_rchild()) //递归右子树  tmp.bHasSibling = TRUE  **TREE-SERIAL**(p->get\_rchild(), files)  **else**  tmp.bHasSibling = FALSE |

#### 从序列化的文件树中恢复二叉文件树（递归）

|  |
| --- |
| **REBUILD-TREE(root, files)**  Inode tmp = files.front()  files.pop\_front()  root->set\_inode(tmp)  **if** (tmp.bHasChild) //递归建立左子树  root->new\_lchild()  root = root->get\_lchild()  **REBUILD-TREE**(root, files)  **if** (tmp.bHasSibling) //递归建立右子树  root->new\_rchild()  root = root->get\_rchild()  **REBUILD-TREE**(root, files)  root = root->get\_rchild() |

#### 解压文件（递归）

|  |
| --- |
| **EXTRACT-ALL\_FILES(root, ifp, path)**  extract\_file(root->get\_inode(), ifp, path); //解压当前结点的文件  CString tmp = path;  **if** (root->get\_inode().bIsDir)  tmp += root->get\_inode().lpszFileName + "\\"  **if** (root->has\_lchild())  **EXTRACT-ALL\_FILES**(root->get\_lchild(), ifp, tmp)//解压子文件夹  **if** (root->has\_rchild())  root = root->get\_rchild()  **EXTRACT-ALL\_FILES**(root, ifp, path) //解压同层文件 |

#### 读写文件

|  |
| --- |
| **READ-WRITE(filename)**  char buffer[512]  UINT nLen  CFile ifile  if (ifile.Open(filename, CFile::modeRead | CFile::shareDenyWrite,0))  **while** (nLen = ifile.Read(buffer, 512))  ofp->Write(buffer, nLen) |

#### 选择文件夹对话框（BROWSEINFO）

|  |
| --- |
| **FOLDER-SELECT()**  BROWSEINFO bi  ZeroMemory(&bi,sizeof(BROWSEINFO))  LPMALLOC pMalloc  LPITEMIDLIST pidl = SHBrowseForFolder(&bi)  TCHAR \* pth = new TCHAR[MAX\_PATH]  if(pidl != NULL)  SHGetPathFromIDList(pidl,pth)  if(SUCCEEDED(SHGetMalloc(&pMalloc)))//pidl指向的对象用完应该释放  pMalloc->Free(pidl)  pMalloc->Release()  CString path(pth)  pDoc->build\_dir\_tree(path)  pDoc->UpdateAllViews(0,0,0) |

#### 显示文件树中的文件夹到ListVew（递归）

|  |
| --- |
| **SHOW-FILES(pRoot, num, index, level)**  LVITEM lvitem  lvitem.mask |= LVIF\_INDENT  lvitem.iIndent = level  for (UINT i=0; pRoot; i++)  if (num == index) //处理双击事件  pRoot->is\_opened = !pRoot->is\_opened  lvitem.iItem = num++  CreateItem(pRoot->get\_inode(), lvitem, pRoot->is\_opened)  if (pRoot->is\_opened)  ShowFiles(pRoot->get\_lchild(), num, index, level+1)//子文件夹  pRoot = pRoot->get\_rchild() |

### 压缩文件结构及打包/解包步骤

#### 压缩文件结构

压缩文件包含三个区域：文件头、数据区、结构区文件头固定4字节，包含一个UINT，为结构区的相对文件头的偏移量；数据区按文件树先序遍历的顺序存放文件，没有多余信息。数据区放到结构区前面方便添加文件，因为结构区较小，而数据区较大，移动比较耗时。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **文件头** | **数据区** | | | | | | **结构区** | | | | |
| UINT  m\_nTreeStart | 文  件  1 | 文  件  2 | 文  件  3 | 文  件  4 | …… | 文  件n | 文件总数目  UINT n | Inode 1 | Inode 2 | …… | Inode n |

#### 打包步骤

* 预留出文件头的四个字节
* 先序遍历文件树，将要打包的文件写入数据区，并将Inode序列化到数组；
* 数据区写入完成后，将文件指针的位置写入文件头
* 在结构区开头记录总文件数
* 将Inode数组写入结构区

#### 解包步骤

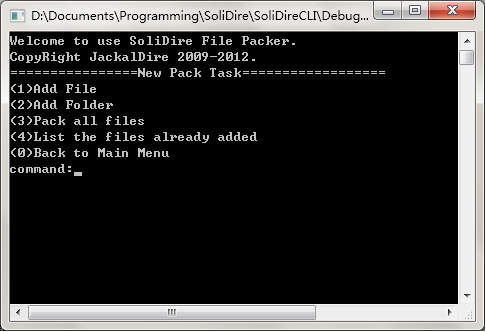
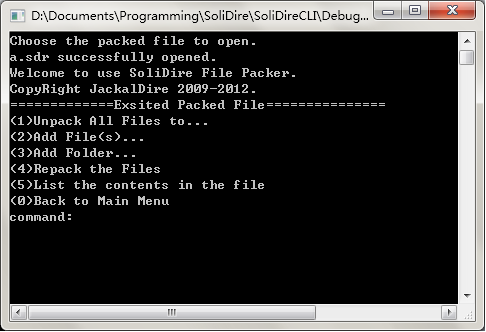
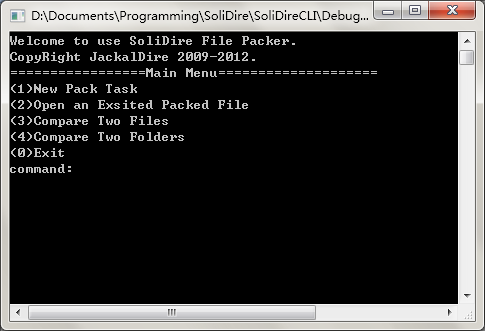
* 读文件头，得到结构区偏移量m\_nTreeStart
* 移动文件指针到结构区，读出文件总数n
* 循环n次读出所有inode到数组
* 从inode数组恢复出文件二叉树

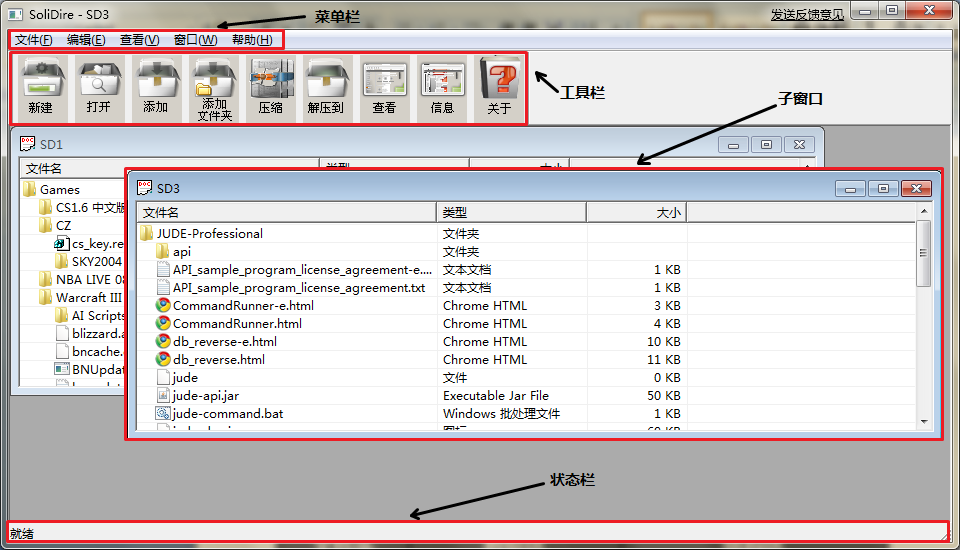
### CLI命令行界面说明

CLI采用多级菜单，见图

### GUI图形界面说明

采用mfc muti-document界面，由标题栏、菜单栏、工具栏、状态栏和子窗口构成。如图：





### 尚未解决的问题

* 无法在任意位置添加文件，只能添加文件到根目录
* 无法删除已经打包的文件