# 需求分析

## 程序功能描述

设计一个简单的带图形界面的文件系统，用文件模拟磁盘，用数组模拟缓冲区，功能如下：

（1） 支持多级目录结构，支持文件的绝对读路径；

（2） 文件的逻辑结构采用流式结构，物理结构采用链接结构中的显式链接方式；

（3） 采用文件分配表 FAT；

（4） 实现的命令操作包括建立目录、列目录、删除空目录、建立文件、删除文件、显示文件内容、打开文件、读文件、写文件、关闭文件、改变文件属性。

## 开发与运行环境

* + 1. **开发语言**

由于题目要求使用图形界面，因此本程序选用C#作为程序设计语言，并使用Windows窗体进行图形界面的设计。

* + 1. **开发环境**

Windows10，Visual Studio 2015，.Net Framework 4.6 64bit。

* + 1. **运行环境**

可在已安装Microsoft .Net Framework 4.6 64位版本的Windows系统上运行。

## 输入、输出形式

类似于普通的Windows窗口程序，对于文件及目录的逻辑操作使用鼠标与程序提供的菜单和按钮进行交互，对于重命名、文本输入等需要手动输入数据的操作可使用键盘键入数据。

## 数据规范

对于数据的规范如下：

* + 1. **模拟磁盘的规格**
       1. 模拟磁盘共有128个物理块，每个物理块大小为64字节。
       2. 磁盘的块从0号开始，到127号结束，FAT（文件分配表）位于0块和1块。
       3. FAT大小为128字节，共有128个项，每个项占用1字节。在FAT的项中，使用“0”（0x00）表示该项对应的块为空闲，使用“-1”（0xFF）表示该项对应的块已结束，使用“1-128”（0x01-0x80）之间的任意一个整数表示该项对应的块的下一块应为上述标记的块号所对应的块。
    2. **目录规范**
       1. 一个目录的目录项信息占用一个块存储，一个目录项信息为8字节，因此一个目录最多只允许有8个子目录（8×8byte = 64byte，为一个盘块的大小）。
       2. 根目录的目录项信息位于2块，根目录名字用“/”表示，表示目录项的块固定于2号，根目录本身的目录项信息不占用磁盘空间，只在内存上存储。
       3. 一个目录项为8字节，每字节的定义如下：

0 2 3 4 5 6 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 目录名 | 0x00 0x00 | 目录属性 | 起始盘块号 | 0x00 |

1. 目录名：用于表示这个目录项对应的目录的名称，占3个字节（实验中合法目录名仅可以使用字母、数字和除 “/”以外的使用ASCII编码的字符，长度不应超过3个字符）；

一个目录在同一父目录下的唯一标识由[目录名]表示，同一目录不允许使用相同标识的多个子目录存在。

1. 保留 2 字节未使用（在磁盘中该项用“0”（0x00）填充）；
2. 目录属性：用于表示这个目录的属性，1个字节，在实际使用中目录属性仅有一个，使用“8”（0x08）填充，用于表示该项的属性为“目录”；
3. 起始盘块号：用于表示该目录项对应目录的子目录/文件的登记项存储的盘块号，占1个字节；
4. 保留 1 字节未使用（在实验中填写“0”（0x00））。
   * 1. **文件规范**
        1. 一个文件项信息为8字节，因此一个目录最多只允许有8个子文件（8×8byte = 64byte，为一个盘块的大小）。
        2. 一个文件项信息为8字节，每字节的定义如下：

0 2 3 4 5 6 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文件名 | 文件类型 | 文件属性 | 起始盘块号 | 文件占用块数 |

1. 文件名：用于表示这个文件项对应的文件的名称，占3个字节（实验中合法文件名仅可以使用字母、数字和除 “/”以外的使用ASCII编码的字符，长度不应超过3个字符）；
2. 文件类型：用于表示这个文件项对应的文件的类型名称，可类比于Windows系统中文件的扩展名信息，占2个字节（实验中合法文件类型名仅可以使用字母、数字和除 “/”以外的使用ASCII编码的字符，长度不应超过2个字符）；

一个文件在同一目录下的唯一标识由[文件名].[文件类型]表示，同一目录不允许使用相同标识的多个文件存在。

1. 文件属性：用于表示这个文件的属性，1个字节，在实际使用中文件属性的标记与含义如下：

“只读”属性：代表该文件为只读属性，标记为“1”（0x01）；

“普通”属性：代表该文件为可读写属性，标记为“4”（0x01）；

“系统”属性：代表该文件为系统属性，标记为“2”（0x01）；

1. 起始盘块号：用于表示该文件项存储实际内容的第一个盘块号，占1个字节；
2. 文件占用盘块数：用于表示该文件的大小，占1个字节。由于要在1个字节内表示文件大小，因此该标识由文件包含的内容占用磁盘存储的块数来表示，在实际程序上显示文件大小时由该值乘以64（一个块的大小，单位：字节）来表示。

## 实现功能介绍

该模拟磁盘文件系统实现了题目要求的功能，详细介绍如下：

* + 1. **目录操作**
       1. 新建子目录

在新建子目录时，需指定目录的名称，确认后程序会检查当前父目录是否有与指定名称相同的子目录，若出现相同则提示失败，若没有相同目录则会在当前目录内创建一个空的子目录，该目录的名称为用户指定的名称。

* + - 1. 显示目录内容

用户可仿照Windows文件管理器操作，通过双击目录的图标进入该目录，程序则会自动列出该目录下包含的子目录和文件的图标与信息。

* + - 1. 删除空目录

用户可通过右键点击目录->选择“删除”来删除该目录。当选择删除时，程序会检测该目录是否为空，若不为空则提示删除失败。若为空，则从当前目录内删除选定的目录。

* + - 1. 显示当前目录路径

程序会自动显示出当前目录的绝对路径，每个目录之间以“/”字符隔开，当切换当前目录时，绝对路径的显示会随即发生变化。

* + - 1. 切换当前目录（通过相对路径）

程序会自动将当前目录下的子目录通过图标的方式显示在屏幕上，要切换到当前目录下的某个子目录，只需双击要进入的目录即可。

* + - 1. 切换当前目录（通过绝对路径）

程序提供通过输入要进入目录的绝对路径，跳转到指定的目录的功能。要输入路径，用户可以用鼠标点击程序上显示当前目录路径的位置，程序会弹出输入绝对路径的文本框，用户输入完路径后，程序会检测用户输入的路径是否合法，若不合法则提示错误。若输入为合法路径，程序会将当前目录切换到用户指定的目录，并显示出该目录下的子目录和文件图标。

* + 1. **文件操作**
       1. 新建空文件

在新建文件时，需指定文件的名称、类型名和属性（普通（即读写）、只读、系统），确认后程序会检查当前父目录是否有与指定名称和类型名同时相同的文件，若出现相同则提示失败，若没有相同文件则会在当前目录内创建一个空的子文件，该文件的名称和类型名为用户指定的名称和类型名，属性为用户指定的属性。

* + - 1. 打开文件

只需双击要打开的文件，该文件将进入“打开”状态。在双击文件时，程序会检查该文件是否已经被打开，若已打开则提示错误，否则文件随即进入“打开”状态。用户可打开不限数量的文件，但相同的文件用户只允许打开一次。

* + - 1. 显示文件内容

在文件进入“打开”状态后，程序会新建一个窗口，并在里面显示出文件包含的内容。

* + - 1. 修改文件内容

若打开的文件为“只读”文件，用户将无法修改文件内的内容。若为其他属性，则用户可以在显示文件内容的窗口内，修改文件的内容。

* + - 1. 保存文件修改后的内容

当程序检测到文件内容被改动后，则允许文件被保存。用户可以点击窗口内的“保存”按钮，保存当前修改的内容。用户也可以关闭窗口，在程序询问是否保存文件时点击“确认”，系统将会自动保存文件。

* + - 1. 放弃保存文件修改后的内容

若要放弃保存，则可以直接关闭文件，在程序询问是否保存文件时点击“取消”，系统将会放弃保存文件修改的内容，并关闭文件。

* + - 1. 显示文件信息

在目录内右键点击要显示的文件->选择“属性”按钮，程序将显示出该文件的信息，如文件名、类型、属性等。

* + - 1. 修改文件属性

在目录内右键点击要修改的文件->选择“修改文件属性”按钮，程序将允许用户选择一个新的文件属性（包括普通、只读、系统的其中一个），确认后原文件的属性将被修改。

* + - 1. 删除文件

在目录内右键点击要删除的文件->选择“删除”按钮，程序将询问用户是否删除文件，若选择“确认”，程序将删除文件，同时将文件的文件项信息和内容从磁盘上抹去。若选择“取消”，将放弃删除。

# 概要设计

## 项目结构

整个程序包含在一个Visual Studio解决方案“DiskOperationSystem”中，命名空间为DiskOperationSystem，所有类和变量都位于该命名空间中。

## 程序模块定义

* + 1. **总览**

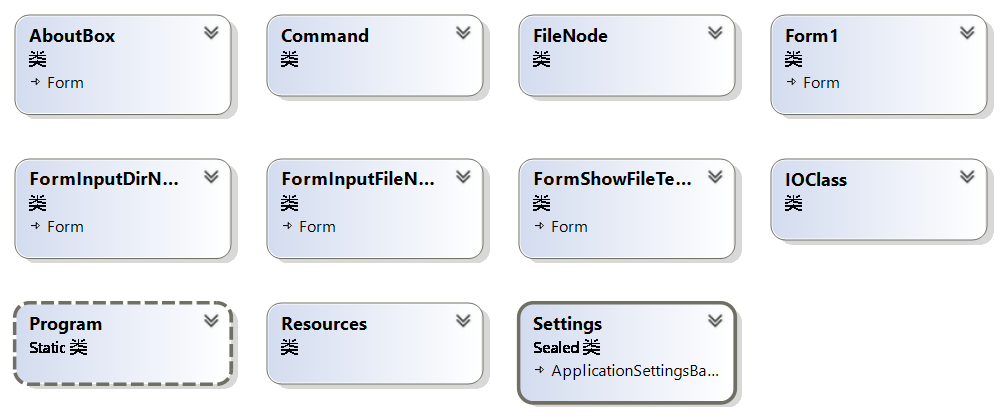


图 1项目总览

* + 1. **类定义**
       1. **Program类**

**说明**

Program类只包含Main方法，是整个应用程序的主入口点。

**UML图**



图 2 Program类

**类包含的方法**

static void Main()

描述：应用程序的主入口点，程序执行将从此方法开始。

返回值：无

* + - 1. **FileNode类**

**说明**

FileNode类定义了一个存储目录项/文件项的基本信息的节点，FileNode类的对象即为一个文件（目录）的项信息在内存中的表示。

**UML图**



图 3 FileNode类

**类的成员变量**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 变量名 | 类型 | 数据域 | get/set访问器 | 描述 |
| name | string | private | Name | 项名称 |
| type | string | private | Type | 项类型名 |
| attribute | string | private | Attribute | 项属性 |
| startnode | byte | private | StartNode | 项起始盘块号 |
| length | byte | private | Length | 项大小 |

表 1 FileNode类成员变量

**方法**

public FileNode()

描述：无参构造方法，构建一个默认节点内容为空的节点对象

传入参数：无

返回值：无

public FileNode(ref string \_name, ref string \_type, ref string \_attribute, ref byte \_startnode)

描述：有参构造方法，传入节点内容，构造对应内容的节点对象

传入参数：

返回值：无

* + - 1. **IOClass类**

**说明**

IOClass类包含对磁盘模拟文件disk进行读写操作的静态方法，提供给整个命名空间内的类和方法调用。

IOClass类的方法封装了对disk文件进行字节级别的最基本读写操作，在IOClass类以外的所有对于磁盘的读写操作均通过调用IOClass类内的方法组合而成，不独自对文件进行二进制I/O操作，以保证程序结构的层次性。

**UML图**



图 4 IOClass类

**成员变量**

|  |  |  |  |
| --- | --- | --- | --- |
| 变量名 | 类型 | 数据域 | 描述 |
| diskFilePath | string | private | 表示磁盘文件路径的字符串 |

表 2 IOClass类成员变量

**方法**

public static void FileWrite(int offset, ref byte[] buffer)

描述：

在磁盘文件disk内从指定位置开始，根据指定的缓冲区的内容，以字符为单位写入磁盘文件中

传入参数：

int offset：要写入的位置相对于磁盘开头的偏移量，从0开始，以字节为单位

ref byte[] buffer：缓冲区byte数组的引用，写入磁盘的数据将以字节为单位存储于该缓冲区中

返回值：

无

public static void FileRead(ref byte[] buffer, int offset, int numOfByte)

描述：

在磁盘文件disk内从指定位置开始，以字符为单位读取多个字符，并存放进缓冲区中

传入参数：

ref byte[] buffer：缓冲区byte数组的引用，读取的数据将以字节为单位存储于该缓冲区中

int offset：要读取的位置相对于磁盘开头的偏移量，从0开始，以字节为单位

int numOfByte：要读取并存放进缓冲区的字节数

返回值：

无

* + - 1. **Command类**

**说明**

Command类封装了对磁盘进行基本原子操作的方法，这些方法通过调用IOClass类的二进制输入/输出方法，对上层调用隐藏了底层二进制I/O的过程，上层方法可通过直接调用该类的方法，实现新建项、删除项、获取块内容等基本操作，亦可通过对类内方法进行组合，实现更复杂的功能。

**UML图**



图 5 Command类

**方法**

public static void md(ref string name)

描述：

根据指定的目录名，新建一个子目录到当前目录下

传入参数：

ref string name：要新建的目录的名称的引用

返回值：

无

public static void newFile(ref string name, ref string type, ref string attribute)

描述：

根据指定的文件名、文件类型、文件属性，新建文件到当前目录下。

传入参数：

ref string name：要新建的文件的文件名的引用

ref string type: 要新建的文件的类型的引用

ref string attribute：要新建的文件的属性的引用

返回值：

无

public static void delete(ref FileNode node)

描述：

删除当前目录内FileNode指定的子文件或子目录。

传入参数：

ref FileNode node：要删除的文件节点信息

返回值：

无

public static void getFileText(ref string text, ref byte startNodeNum)

描述：

读取从指定的起始盘块号开始的整个文本文件的字符串，只限于读取文本文件。

传入参数：  
ref string text：用于存放读取出的内容的字符串引用

ref byte startNodeNum：需要读取的文本文件的起始盘块号的引用

返回值：

无

public static byte saveFileText(ref string text, ref byte startNodeNum)

描述：

指定要写入的文件内容和该文件的起始盘块号，将内容写入磁盘中，并返回修改后的文件占用的盘块数量。

传入参数：

ref string text：要写入的文本字符串的引用

ref byte startNodeNum：要写入的文件的起始盘块号的引用

返回值：

返回修改后的文件占用的盘块数量

* + - 1. **Form1类**

**说明**

Form1类继承于System.Windows.Forms，类内描述了程序的主窗体包含的控件、视图、响应事件等，该类的对象用于程序主窗口界面的显示，在程序初始化时被创建。

**Using程序集引用**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using Microsoft.VisualBasic;

**窗体及其控件说明**



图 6 Form1类

|  |  |  |
| --- | --- | --- |
| 控件名及说明 | 事件类型 | 事件及说明 |
| 目录ToolStripMenuItem1  菜单栏->新建->目录 | Click | 目录ToolStripMenuItem1\_Click()  执行后响应“新建目录”事件 |
| 文件ToolStripMenuItem2  菜单栏->新建->文件 | Click | 文件ToolStripMenuItem2\_Click()  执行后响应“新建文件”事件 |
| 大图标ToolStripMenuItem  菜单栏->视图->大图标 | Click | 大图标ToolStripMenuItem\_Click()  执行后将当前视图改为“大图标”视图 |
| 小图标ToolStripMenuItem  菜单栏->视图->小图标 | Click | 小图标ToolStripMenuItem\_Click()  响应后将当前视图改为“小图标”视图 |
| 平铺ToolStripMenuItem  菜单栏->视图->平铺 | Click | 平铺ToolStripMenuItem\_Click()  响应后将当前视图改为“平铺”视图 |
| 列表ToolStripMenuItem  菜单栏->视图->列表 | Click | 列表ToolStripMenuItem\_Click()  响应后将当前视图改为“列表”视图 |
| 详细信息ToolStripMenuItem  菜单栏->视图->详细信息 | Click | 详细信息ToolStripMenuItem\_Click()  响应后将当前视图改为“详细信息”视图 |
| 关于ToolStripMenuItem  菜单栏->关于 | Click | 关于toolStripMenuItem\_Click()  响应后将弹出“关于”窗口 |
| buttomUpDir  工具栏->上一级目录 | Click | buttonUpDir\_Click()  响应后将当前目录改为上一级，并刷新视图 |
| labelCurrentDir  工具栏->当前路径标签 | Click | labelCurrentDir\_Click()  响应后将提示用户输入绝对路径并跳转 |
| mainListView  文件列表视图 | DoubleClick | mainListView\_DoubleClick()  响应后将执行打开文件/进入目录操作 |
| mainContextMenuStrip  视图内的右键菜单 | Opening | mainContextMenuStrip\_Opening()  响应后将打开右键菜单，  并根据选定的控件类型显示对应的菜单项 |
| Closed | mainContextMenuStrip\_Closed()  响应后将关闭右键菜单，  并将所有菜单项可视状态重置为默认 |
| 视图toolStripMenuItem1  右键菜单->视图 |  | 与菜单栏->视图菜单内各控件的响应事件相同 |
| 新建ToolStripMenuItem  右键菜单->新建 |  | 与菜单栏->新建菜单内各控件的响应事件相同 |
| 重命名toolStripMenuItem  右键菜单->重命名 | Click | 重命名toolStripMenuItem\_Click()  响应后将执行重命名操作 |
| 删除toolStripMenuItem  右键菜单->删除 | Click | 删除toolStripMenuItem\_Click()  响应后将执行删除选中的文件/目录的操作 |
| 修改文件属性toolStripMenuItem  右键菜单->修改文件属性 | Click | 修改文件属性toolStripMenuItem\_Click()  响应后将执行修改文件的属性的操作 |
| 刷新toolStripMenuItem  右键菜单->刷新 | Click | 刷新toolStripMenuItem\_Click()  响应后将刷新列表的视图 |
| 属性ToolStripMenuItem  右键菜单->属性 | Click | 属性toolStripMenuItem\_Click()  响应后将执行文件的属性 |

表 3 Form1类成员变量

* + - 1. **FormInputDirName类**

**说明**

该Windows窗体类设计了一个用于提供用户输入目录名称信息的对话框式界面。该窗体对象可在程序需要用户提供目录名称的时候被创建，如新建目录、重命名目录操作进行中时。程序可以根据使用场景的不同在该提示框内显示不同的提示语句。

**Using程序集引用**

using System;

using System.Windows.Forms;

**窗体及其控件说明**



图 7 FormInputDirName类

|  |  |
| --- | --- |
| 控件名 | 说明 |
| label1 | 显示该提示框的提示消息 |
| buttonOK | “确认”按钮，点击后将提交信息 |
| textBoxInput | 用于用户输入目录名称的文本框 |

表 4 FormInputDirName类成员变量

* + - 1. FormInputFileName**类**

**说明**

该Windows窗体类设计了一个用于提供用户输入文件名称、类型、属性信息的对话框式界面。该窗体对象可在程序需要用户提供文件信息的时候被创建，如新建文件、重命名文件、修改文件属性操作进行中时。程序可以根据使用场景的不同在该提示框内显示不同的提示语句和控件，如在修改属性时关闭文本框显示的操作。

**Using程序集引用**

using System;

using System.Windows.Forms;

**窗体及其控件说明**



图 8 FormInputFileName类

|  |  |
| --- | --- |
| 控件名 | 说明 |
| label提示 | 显示该提示框的提示信息 |
| textBoxName | 用于输入文件名的文本框 |
| 文件名label | 指示该处textBox应该输入文件名 |
| checkBox普通  checkBox只读  checkBox系统 | 用于用户勾选文件属性为“普通”、“只读”或“系统 |
| label属性 | 指示该处checkBox应勾选文件属性 |
| buttom确定 | 确认按钮，点击后提交信息 |

表 5 FormInputFileName类成员变量

* + - 1. FormShowFileText**类**

**说明**

该Windows窗体类的对象用于提供用户对程序内文件的交互操作。对象可在用户打开文件时被建立，在初始化对象时，程序会读取文件内的文本内容，并在对象内显示出来，以供用户的后续操作。用户可在该窗体对象的操作包括：显示文本内容、修改文本内容、保存修改后的文本内容等。

**Using程序集引用**

using System;

using System.Linq;

using System.Text;

using System.Windows.Forms;

**窗体及其控件说明**



图 9 FormShowFileText类

|  |  |
| --- | --- |
| 控件名 | 说明 |
| label1 | 指示textBox为显示文件内文本内容的文本框 |
| textBox1 | 用于显示文本内容的文本框 |
| labelCurrentTextNum | 指示文本框内修改后的总字符数 |
| buttomSave | 保存按钮，点击后提交编辑文本内容 |

表 7 FormShowFileText类成员变量

* + - 1. AboutBox**类**

**说明**

该Windows窗体类只用于提供程序“关于”信息的显示，内含作者、版权信息等说明，不包含交互操作。

## 程序模块间的层次关系



图 10 项目层次关系

自底向上，本程序由四个层次组成：

第1层：磁盘文件

该层位于程序的最底层，使用一个二进制文件模拟了一个使用FAT分区表格式的磁盘，容量为8KB，所有的数据在文件内均由二进制数0、1表示。

第2层：IOClass类

该类模拟了真实磁盘中磁头对磁盘的寻址、读写操作，封装了对磁盘文件的读写方法，提供了原始字节级别的对磁盘文件的读写操作，与上层的数据交互通过byte[]数组传递。

第3层：Command类

该类内含程序要完成复杂任务所需的原子操作，通过调用IOClass类的磁盘I/O方法，获取一个原子操作所需的信息，并对磁盘文件进行修改。上层方法通过对Command类内方法的不同组合，完成用户要求的复杂任务。

第4层：窗体、对话框

该层位于程序的最高层，由Windows窗体、控件及控件的响应事件组成。用户通过该层与程序进行交互操作，该层负责为用户的输入/输出产生不同的响应事件，通过对Command类内方法的调用，最终完成复杂的任务，并实时向用户反馈结果。

## 程序流程描述

* + 1. **主界面执行流程**



图 11 主流程

* + 1. **程序子功能执行流程**
       1. **新建目录**

图 12 新建目录流程

* + - 1. **切换子目录**



图 13切换子目录流程

* + - 1. **通过绝对路径访问目录**



图 14通过绝对路径访问目录流程

* + - 1. **重命名目录/文件**



图 15重命名流程

* + - 1. **删除目录/文件**



图 16删除流程

* + - 1. **新建文件**



图 17新建文件流程

* + - 1. **打开文件**



图 18打开文件流程

* + - 1. **关闭已打开的文件**



图 19保存文件流程

# 详细设计

以下介绍了程序内各个主要方法的设计思路，使用文字及伪代码描述。

为简明扼要的说明程序的工作方式和原理，下面仅对复杂度较高的方法列出用伪代码描述的工作思路，对简单的方法块仅使用文字描述。

## 基本输入输出——IOClass类

* + 1. **往磁盘文件写入二进制数据：FileWrite()方法**

**描述**

在磁盘文件disk内从指定位置开始，根据指定的缓冲区的内容，以字节为单位写入磁盘文件中。

**伪代码**

public static void FileWrite(int offset, ref byte[] buffer)

{

使用File.OpenWrite返回一个打开方式设置为 OpenOrCreate，访问方式设置为Write的FileStream对象，并被BinaryWriter引用

设置指针位置为从磁盘文件开头起偏移offset偏移量的位置

将buffer内的数据写进磁盘文件的该位置

写入结束，关闭流

}

* + 1. **从磁盘文件读取二进制数据：FileRead()方法**

**描述**

该方法在磁盘文件disk内从指定位置开始，以字节为单位读取多个字符，并存放进缓冲区中。

**伪代码**

public static void FileRead(缓冲区的byte数组, 偏移量, 要读取的字节数)

{

使用File.OpenRead返回一个访问方式设置为Read的FileStream对象，并被BinaryReader引用

设置指针位置为从磁盘文件开头起偏移offset偏移量的位置

从该位置开始读取数据进入缓冲区

读取结束，关闭流

}

## 磁盘操作函数——Command类

* + 1. **新建目录：md()方法**

**描述**

根据指定的目录名，写入一个新的目录项到当前目录下，并为新目录分配一个盘块。

**伪代码**

public static void md(要新建的目录的字符串)

{

新建buffer缓冲区，用于存放父目录的节点的信息

获得当前目录的信息，用于从磁盘索引子目录项

判断是否重名

for 一个目录内的8个子项

{

对于第i个项，比较要新建的目录与第i项的目录是否同名

if (第i项为目录 && 该项与要新建的目录同名)

{

提示新建错误，返回系统

}

}

判断该目录内是否存在可以存放要新建的目录项的空间

如果有，随即写入

for 一个目录内的8个子项

{

判断当前第i项是否为空

if (当前第i项为空项)

{

构造一个8字节的目录项信息，并放入buffer缓冲区中

遍历FAT表，获得空的盘块并写入起始盘块号

调用FileRead()读取磁盘FAT表存放到内存

for FAT表的128个项

{

if 当前项j为空

{

将j记载为目录的起始盘块

修改FAT表，写入j项为最后一项，调用FileWrite()写入

将项信息写入父目录的块中，调用FileWrite()写入

调用FileWrite()将存放构造的项信息写入对应位置

新建结束

}

}

如果当前项j不为空，继续寻找下一个空块

}

}

}

* + 1. **新建文件：newFile()方法**

**描述**

根据指定的文件名、文件类型、文件属性，新建文件到当前目录下。

**伪代码**

public static void newFile(新文件的名称, 新文件的类型名, 新文件的属性)

{

新建buffer缓冲区，用于存放父目录的块内包含索引的信息

获得当前目录的信息，用于从磁盘索引子目录项

调用FileRead()，将父目录的块信息读取到buffer中

根据获取的块内的信息，判断要新建的文件是否与当前存在的文件重名

for 目录内的8个子项

{

对于第i个项，比较要新建的文件与第i项的文件是否同名

if (第i项为文件 && 该项与要新建的文件同名)

{

提示新建错误，返回系统

}

}

没有发生重名，即将开始写入索引

下面判断block内是否有可以存放node的空间并写入

for 目录内的8个子项

{

判断当前第i项的位置是否为空项

if 当前第i项为空

{

根据新建文件时填写的信息，构造一个8字节的文件项，并放入数组s中

新建fatBytes的byte数组，调用FileRead()临时存放磁盘FAT表到内存

下面开始遍历FAT表，获得空的盘块并写入起始盘块号

for fat的128项中的第j项

{

if 当前第j项所指的块为空块

{

将该块登记为文件的起始盘块，写入文件项的起始盘块号中

修改FAT表，将该块的索引填写为最后一块的标记，使用FileWrite()写入到磁盘中

将已经构造好的文件项信息写入父目录的块中

写入结束

}

}

若第j项不为空，继续寻找下一个FAT索引

}

}

}

* + 1. **删除文件及目录：delete()方法**

**描述**

删除当前目录内由FileNode对象指定的子文件或子目录。

**伪代码**

public static void delete(ref FileNode node)

{

从当前父目录的块中找到要删除的文件 / 目录的索引，并删除，用'0'填充

从FAT表中将待删除文件 / 目录所占用的块信息用'0x00'填充

过程如下：

将磁盘的FAT表加载到内存

记录要删除节点的起始盘块号

if (要删除的目录 / 文件只有一块)

{

将FAT表内对应块的标记信息改为'0x00'

从磁盘内删除对应块内的信息

删除成功，提示信息，返回主界面

}

当要删除的目录 / 文件占有多个块的情况，根据FAT表记载的信息迭代删除

while (当前删除的块不是文件的最后一个块)

{

保留当前操作的块的块号，防止被修改后丢失

删除当前块的内容

从FAT表找到当前块的下一个块，并将当前操作块改成索引指示的块，同时将前一个块的FAT信息设为'0x00'

if (删除到最后一个块)

按照上面的“要删除的目录 / 文件只有一块”操作进行

}

}

* + 1. **从文件内获得内容：getFileText()方法**

**描述**

读取从指定的起始盘块号开始的整个文本文件的字符串。

**伪代码**

public static void getFileText(ref string text, ref byte startNodeNum)

{

将整个FAT表载入到内存

通过要读取文件的的起始盘块号，找到文件的起始块

if (当前读取的块为文件的最后一块)

{

读取整个块的内容到存储的字符串中

读取完毕，返回上层方法

}

while (当前读取的块不是文件的最后一块)

{

读取当前整个块的内容，追加到存储的字符串的末尾

根据FAT表，设置当前节点为下一个节点

}

if (已经读取到文件的最后一块)

{

读取整个块的内容，追加到存储的字符串的末尾

读取完毕，返回上层方法

}

}

* + 1. **往文件内保存内容：saveFileText()方法**

**描述**

指定要写入的文件内容和该文件的起始盘块号，将内容写入磁盘中，并返回修改后的文件占用的盘块数量。

**伪代码**

public static byte saveFileText(ref string text, ref byte startNodeNum)

{

初始化局部变量：用于在保存过程中统计文件占用的块的数量

仿照delete()方法，将文件原有的内容删除

下面开始重新写入文件内容

索引FAT表，获得空的盘块并将这个块记为文件的起始盘块号

当要写入的内容长度超过一个盘块（64字节）时每分割64字节迭代存入

while (要写入的内容长度超过64字节)

{

在记录写入内容的字符串中截取前64个字节保存到临时变量中，并从字符串中删除这64字节

将该64字节写入空块中

将这块在FAT表的索引临时记为最后一块，防止下一步寻找空块时出现错误

在FAT表找到下一个空闲块，将当前工作的块修改该块，同时把找到的块的块号写入FAT表上一块的索引中，使上一块链接到该块

}

当写入到最后一个盘块（即text的长度小于64字节时），此时当前的工作块应为文件的最后一个盘块

将这个块的索引信息0xFF写入到FAT表中

将文本内容写入这个块内

写入结束

返回在写入过程中使用的块的数量，用于上层方法记载文件的大小

}

## 主窗体：Form1类

* + 1. **初始化根目录：InitializeCurrentDir()方法**

**描述**

该方法在Form1初始化时被调用，用于将根目录的目录项信息加载到内存，并初始化当前的工作目录为根目录"/"。

**伪代码**

private void InitializeCurrentDir()

{

初始化当前目录为根目录"/"

新建一个FileNode对象，将根目录的文件信息放进FileNode

将其插入表示当前打开目录的List集合（CurrentDirList）的第一项

将指示当前目录的文本提示改为根目录的名称"/"

}

* + 1. **刷新目录内容：updateList()方法**

**描述**

当调用这个方法时，程序会当前目录内的子项，并将其显示在目录列表中，相当于Windows中的“刷新”操作。

**伪代码**

public void updateList()

{

初始化更新操作：

开始List的数据更新，UI线程暂时挂起，停止刷新视图

清除List内的所有元素以重新写入

设定List内元素要引用的图标集

开始更新：

根据当前目录的FileNode对象上记载的信息，读取目录对应的块内容到缓冲区中

根据块上记载的索引新建列表项

for (块上的8个索引)

{

if (该索引对应的位置为空)

{

跳过录入当前位置的信息

}

创建Subitem对象，根据索引记载的内容，分别录入名称、类型名等信息到对象中

将这个对象添加到列表中

}

结束数据处理，恢复UI刷新线程

}

## 输入目录名称的窗体：FormInputDirName类

该类构造的对象可同时用于作为新建目录、重命名目录的对话框。

* + 1. **将对话框提示内容替换成指定的字符串：changeLabelName()**

**说明**

由于该窗口可同时用于新建目录和重命名目录时用于提示用户输入目录名的对话框，因此需要为两个功能分别使用不同的提示语，通过在新建提示框时调用该方法，可以修改对话框内的提示内容的文本为指定的文本。

## 输入文件项信息的窗体：FormInputFileName类

该类构造的对象可同时用于作为新建文件、重命名文件、修改文件属性的对话框。

* + 1. **获取用户输入的文件名称：getFileName()**

**说明**

允许上层方法调用此方法，在对话框关闭时，获取用户指定的名称，用于进一步操作。

* + 1. **获取用户指定的文件属性：getFileAttribute()**

**说明**

允许上层方法调用此方法，在对话框关闭时，获取用户指定的属性值，用于进一步操作。

* + 1. **设置为修改文件属性的布局：isChangeAttribute()**

**说明**

当该窗体对象用于作为修改文件属性的对话框，需要修改对话框的提示内容，隐藏输入文件名的文本框和提示标签，上述动作通过上层方法在构造此窗体时通过调用此方法来实现。

## 显示文件详细内容的窗体：FormShowFileText类

* + 1. **保存用户修改的文件内容：saveFile()**

**说明**

当用户选择保存修改后的文件内容时，程序会调用该方法，在磁盘清空原有的文件内容并归还盘块后，将更新后的文本内容转换为ASCII格式的字符串，以二进制方式写入到磁盘中。