

文件管理项目

**——简单文件管理系统设计方案**

项目说明文档

作 者 姓 名： 刘子赫

学 号： 2250694

指 导 老 师： 张惠娟

专 业： 软件工程

同济大学

Tongji University

目录

[1. 引言 3](#_Toc169453109)

[2. 项目需求 4](#_Toc169453110)

[2.1 基本任务 4](#_Toc169453111)

[2.2 具体要求 4](#_Toc169453112)

[2.3 功能描述 4](#_Toc169453113)

[3. 核心代码介绍 4](#_Toc169453114)

[3.1 simple\_file\_system.py 4](#_Toc169453115)

[3.2 GUI.py 6](#_Toc169453116)

[4. 成果展示 8](#_Toc169453117)

[5. 总结 13](#_Toc169453118)

[5.1 程序优点 13](#_Toc169453119)

[5.2 程序不足 14](#_Toc169453120)

[6. 说明 14](#_Toc169453121)

1. 引言

在当今的数字时代，文件管理是每个计算机用户日常操作的重要组成部分。SimpleFileSystem 旨在提供一个直观且易于使用的界面，使用户能够轻松地管理他们的文件和目录。无论是创建、删除、读取还是写入文件，SimpleFileSystem 都能为您提供便利。该应用程序特别适合学习和演示文件系统操作的基本概念，也适合 python 入门的一个项目，同时也是一个很好的 PyQt 编程示例。

本报告旨在介绍同济大学软件学院操作系统课程的第三个项目——文件管理系统的设计与实现，通过本报告的阐述，读者将文件管理系统有一个全面的了解，并能够理解其在实际应用中的重要性和价值。

1. 项目需求
   1. 基本任务

* 在内存中开辟一个空间作为文件存储器，在其上实现一个简单的文件系统;
* 退出这个文件系统时，需要该文件系统的内容保存到磁盘上，以便下次可以将其回复到内存中来。
  1. 具体要求
* 读取和写入文件文件存储空间管理可采取链接结构（如FAT文件系统中的显式链接等）或者其他学过的方法；
* 空闲空间管理可采用位图或者其他方法；
* 文件目录采用多级目录结构，目录项目中应包含：文件名、物理地址、长度等信息。
  1. 功能描述
* - 创建文件和目录
* - 删除文件和目录
* - 读取和写入文件
* - 显示文件和目录属性
* - 复制和粘贴操作（仅支持文件）

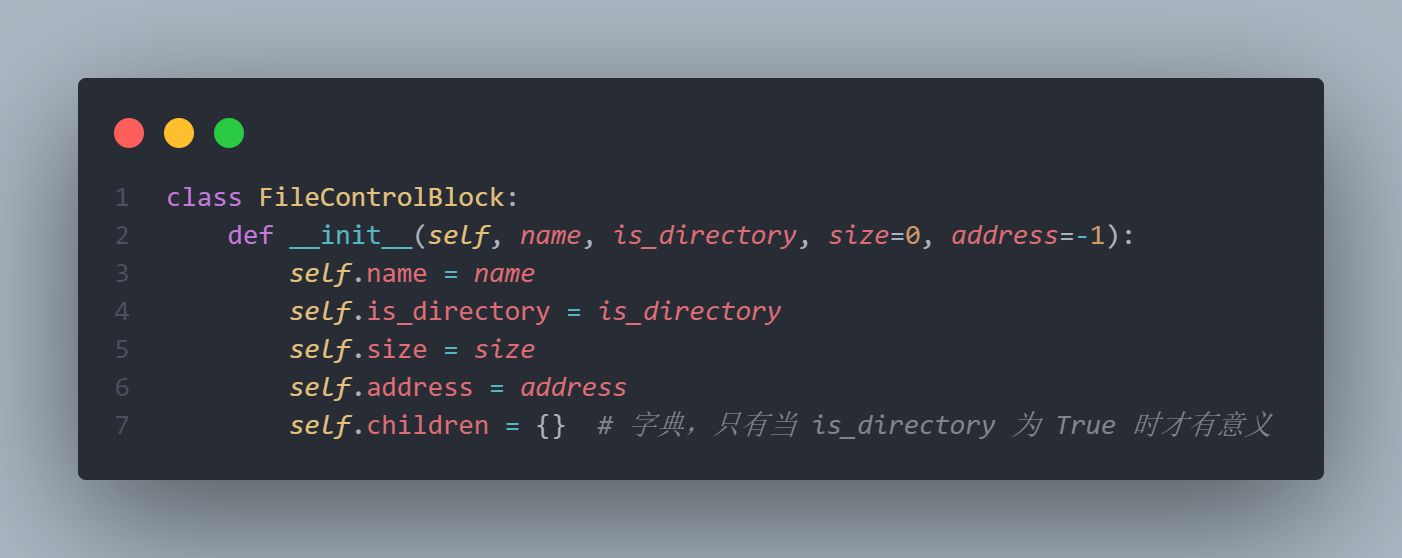
1. 核心代码介绍
   1. simple\_file\_system.py

这部分代码是文件管理系统的底层实现逻辑。实现了一个简单的文件系统的底层逻辑管理，使用FAT文件分配表来管理文件和目录的存储、创建、删除、读取和写入操作。它支持基本的文件操作，如格式化、保存和加载文件系统、分配和释放数据块、以及处理文件和目录的层次结构。

以下是它的核心类和方法：

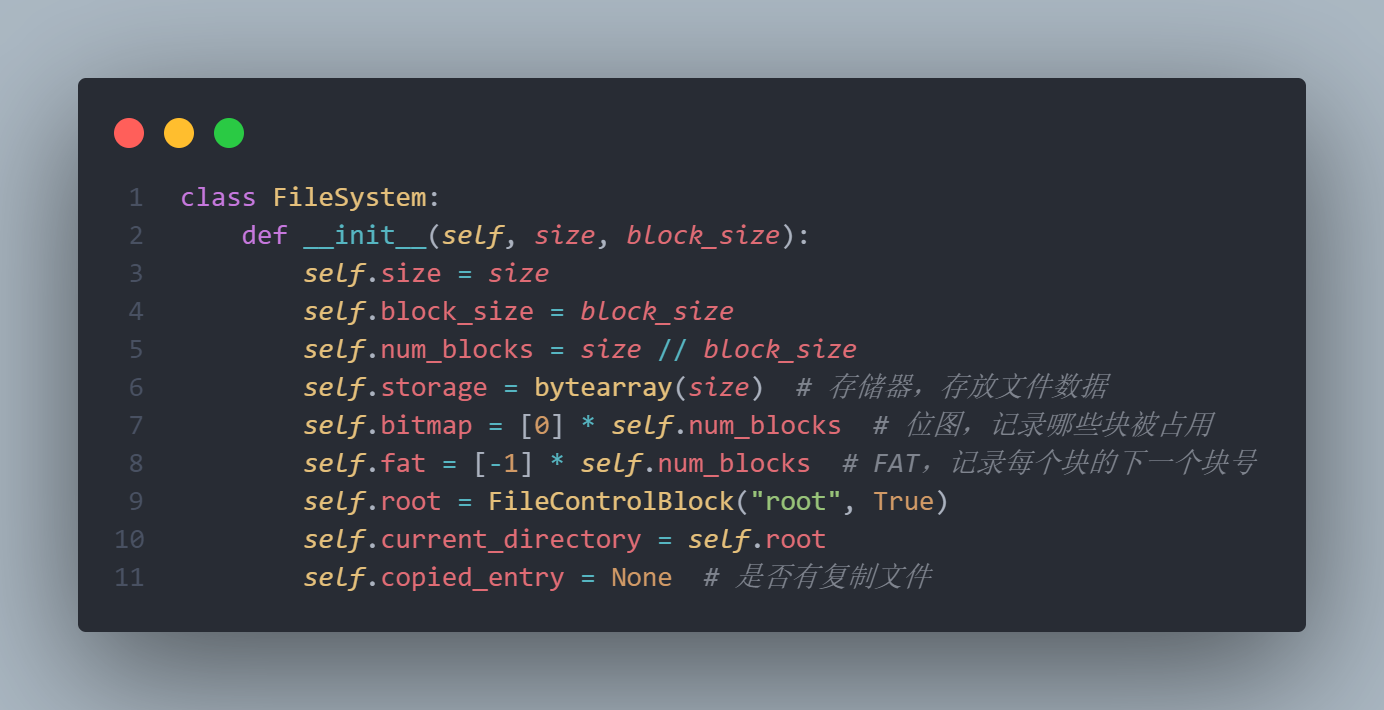
* **FileControlBlock 类**

FileControlBlock 类代表文件控制块（FCB），用于表示文件或目录的信息。



* **FileSystem 类**

文件系统类，管理文件和目录的创建、删除、读取、写入等操作。



该类方法下有多个方法，以下是对各个方法的说明（详细解释见代码）：

format: 初始化文件系统，重置存储器、位图和文件控制块。

save\_to\_disk: 将文件系统保存到磁盘上的文件。

load\_from\_disk: 从磁盘文件加载文件系统。

allocate\_block: 分配一个空闲数据块。

free\_block: 释放一个数据块。

create\_file: 创建一个新文件。

* + 检查文件是否存在： 检查当前目录中是否已经存在同名的文件或目录。
  + 计算所需数据块数： 根据文件大小计算需要分配的数据块数量。
  + 检查空闲块是否足够： 确认位图中是否有足够的空闲块。
  + 分配数据块： 分配所需的数据块，并将它们链接起来。
  + 创建文件控制块（FCB）： 创建一个新的文件控制块，并将其添加到当前目录中。
  + 处理异常情况： 如果在分配数据块过程中发生异常，释放已分配的数据块并返回错误信息。

clear\_file\_data: 清空文件的数据块。

delete\_file: 删除指定文件。

* + 检查文件是否存在： 检查当前目录中是否存在指定名称的文件。
  + 清除剪贴板内容： 如果要删除的文件在剪贴板中，则清除剪贴板。
  + 释放数据块： 遍历文件占用的数据块，释放这些块并清除其中的数据。
  + 删除文件控制块： 从当前目录中删除文件的文件控制块。
  + 处理异常情况： 如果指定名称对应的是目录而不是文件，或文件不存在，打印相应的错误信息。

create\_directory: 创建一个新目录。

delete\_directory: 删除指定目录及其所有内容。

* + 检查目录是否存在： 检查当前目录中是否存在指定名称的目录。
  + 清除剪贴板内容： 如果要删除的目录或其子目录在剪贴板中，则清除剪贴板。
  + 递归删除子目录和文件： 递归遍历目录中的所有子目录和文件，分别调用删除文件或删除目录的方法。
  + 删除目录控制块： 从当前目录中删除指定目录的控制块。
  + 处理异常情况： 如果指定名称对应的不是目录，或目录不存在，打印相应的错误信息。

is\_fcb\_in\_directory: 判断文件控制块是否在指定目录中。

change\_directory: 切换当前工作目录。

open\_file: 打开指定文件。

close\_file: 关闭指定文件。

write\_file: 向指定文件写入数据。

* + 查找文件控制块： 根据路径找到对应的文件控制块。
  + 检查文件类型： 确保目标不是目录，并打开文件。
  + 计算所需数据块数： 根据数据大小计算需要分配的数据块数量。
  + 检查空闲块是否足够： 确认位图中是否有足够的空闲块。
  + 清空原有数据块： 清除文件原有的数据块。
  + 分配数据块： 分配新的数据块，并将它们链接起来。
  + 写入数据： 将数据写入分配的块中。
  + 关闭文件： 关闭文件并更新文件控制块的信息。
  + 处理异常情况： 如果在分配数据块或写入数据过程中发生异常，释放已分配的数据块并返回错误信息。

read\_file: 读取指定文件的数据。

list\_directory: 列出当前目录中的所有文件和目录。

copy\_entry: 复制指定文件或目录。

find\_fcb\_by\_path: 根据路径查找文件控制块。

paste\_entry: 粘贴复制的文件或目录到目标目录。

* + 检查剪贴板内容： 确认剪贴板中有要粘贴的文件或目录。
  + 确认目标目录： 如果指定了目标目录名称，查找该目录控制块；否则使用当前目录。
  + 检查重名冲突： 如果目标目录中已经存在同名文件或目录，为新文件生成唯一的名称。
  + 深度复制文件控制块： 使用深度复制方法将剪贴板中的文件控制块复制到目标目录中。
  + 更新目录结构： 如果粘贴的是目录，递归更新其子目录和文件的父目录引用。
  + 处理异常情况： 如果目标目录无效或不是目录，打印相应的错误信息。

\_update\_fcb\_references: 递归更新文件控制块的父目录引用。

* 1. GUI.py

这部分代码实现了一个基于PyQt5的图形用户界面，用于交互管理自定义文件系统，支持创建、删除、复制、粘贴、读取和写入文件及目录的操作，通过树视图展示文件系统结构，并提供右键菜单和对话框与用户进行交互。

程序默认分配了1MB内存空间，用户也可以调整代码改变内存分配大小。



以下是它的核心方法：

1. populate\_tree：递归填充文件系统树，将文件和目录添加到树视图中。
   * 遍历子项： 对每个子文件或子目录，创建相应的树节点
   * 设置节点数据： 将文件或目录的名称和相关数据（如文件控制块）设置到节点上
   * 递归调用： 如果子项是目录，递归调用 populate\_tree 方法填充其子项。

2. open\_menu：打开右键菜单，提供创建、删除、属性查看等操作选项

* + 获取当前项： 获取用户右键点击位置对应的树节点
  + 创建菜单： 根据节点类型（文件或目录）创建相应的右键菜单
  + 执行操作： 根据用户选择的菜单项执行相应操作，如创建文件、删除目录、查看属性等

3. create\_entry：在当前目录下创建文件或目录。

* + 获取名称： 弹出对话框让用户输入文件或目录的名称
  + 创建文件或目录： 根据用户选择的类型，调用文件系统的相应方法创建文件或目录
  + 更新树视图： 在树视图中添加新创建的文件或目录节点，并展开父节点以显示新项

4. delete\_entry：删除指定文件或目录，并更新树视图。

* + 获取父项： 获取待删除项的父节点和文件控制块
  + 删除文件或目录： 调用文件系统的相应方法删除文件或目录，若为目录则递归删除其内容
  + 更新树视图： 从树视图中移除已删除的节点，如果当前目录被删除，则切换到父目录

5. write\_file：向指定文件写入数据。

* + 获取文件路径： 获取要写入文件的完整路径。
  + 读取现有数据： 调用文件系统方法读取现有文件数据，以便用户在弹出对话框中修改。
  + 弹出输入对话框： 弹出多行文本输入框让用户输入文件内容。
  + 写入数据： 将用户输入的数据写入文件，并显示操作结果信息。

6. read\_file： 读取指定文件的数据，并在对话框中显示。

* + 获取文件路径： 获取要读取文件的完整路径。
  + 读取数据： 调用文件系统方法读取文件数据。
  + 显示数据： 如果文件有数据，将数据在弹出的对话框中显示；否则显示文件为空的信息。

7. copy\_entry：复制指定文件或目录到剪贴板。

* + 获取当前项： 获取当前选择的树节点和对应的文件控制块。
  + 检查有效性： 确认所选项不是根目录。
  + 复制文件或目录： 调用文件系统的 copy\_entry 方法，将文件控制块复制到剪贴板。

8. paste\_entry：将剪贴板中的文件或目录粘贴到当前目录或目标目录。

* + 检查剪贴板内容： 确认剪贴板中有要粘贴的文件或目录。
  + 确认目标目录： 获取当前目录或用户选择的目标目录。
  + 处理重名冲突： 确保粘贴后的文件或目录名称不与目标目录中的现有项冲突。
  + 粘贴文件或目录： 将复制的文件控制块添加到目标目录，并更新树视图。

9. show\_properties：显示文件或目录的属性信息。

* + 获取文件控制块： 获取当前选择的树节点和对应的文件控制块。
  + 创建属性对话框： 创建对话框显示文件或目录的属性信息，如名称、类型、大小（文件）、内容数（目录）。
  + 显示对话框： 弹出属性对话框，让用户查看属性信息。

10. closeEvent：在关闭窗口时，保存文件系统的状态，并确认用户是否要退出。

* + 弹出确认对话框： 弹出对话框询问用户是否确认退出。
  + 保存文件系统： 如果用户确认退出，调用文件系统的 save\_to\_disk 方法保存当前状态。
  + 处理用户选择： 根据用户选择，决定是否关闭应用程序或取消关闭操作。
  + 这些方法涉及复杂的逻辑操作，如递归填充树视图、动态创建和删除文件或目录、处理文件写入和读取等，因此详细说明可以帮助更好地理解其实现细节。

还有一些辅助这些方法的方法：

select\_item: 处理树视图中双击事件，用于选择并读取文件内容。

get\_full\_path: 获取指定文件控制块的完整路径。

find\_parent\_item: 查找指定文件控制块的父节点。

paste\_entry: 将剪贴板中的文件或目录粘贴到目标目录。

change\_directory: 切换当前工作目录。

refresh\_view: 刷新树视图，更新显示文件系统的当前状态。

get\_expanded\_items: 获取当前展开的树节点。

expand\_items: 展开指定名称的树节点。

find\_item: 递归查找树视图中具有指定数据的节点。

find\_item\_by\_fcb: 根据文件控制块递归查找树节点。

find\_item\_by\_name: 根据名称查找树节点。

on\_item\_expanded: 处理树节点展开事件。

on\_item\_collapsed: 处理树节点折叠事件。

show\_properties\_from\_menu: 从右键菜单中显示文件或目录属性。

save\_and\_notify: 保存文件系统并显示通知消息。

create\_file\_in\_current\_directory: 在当前目录创建新文件。

create\_directory\_in\_current\_directory: 在当前目录创建新目录。

create\_entry\_in\_current\_directory: 在当前目录创建新文件或目录。

rename\_entry: 重命名当前选择的文件或目录。

show\_about: 显示关于对话框，介绍文件系统的使用方法。

delete\_directory\_recursively: 递归删除目录及其所有内容。

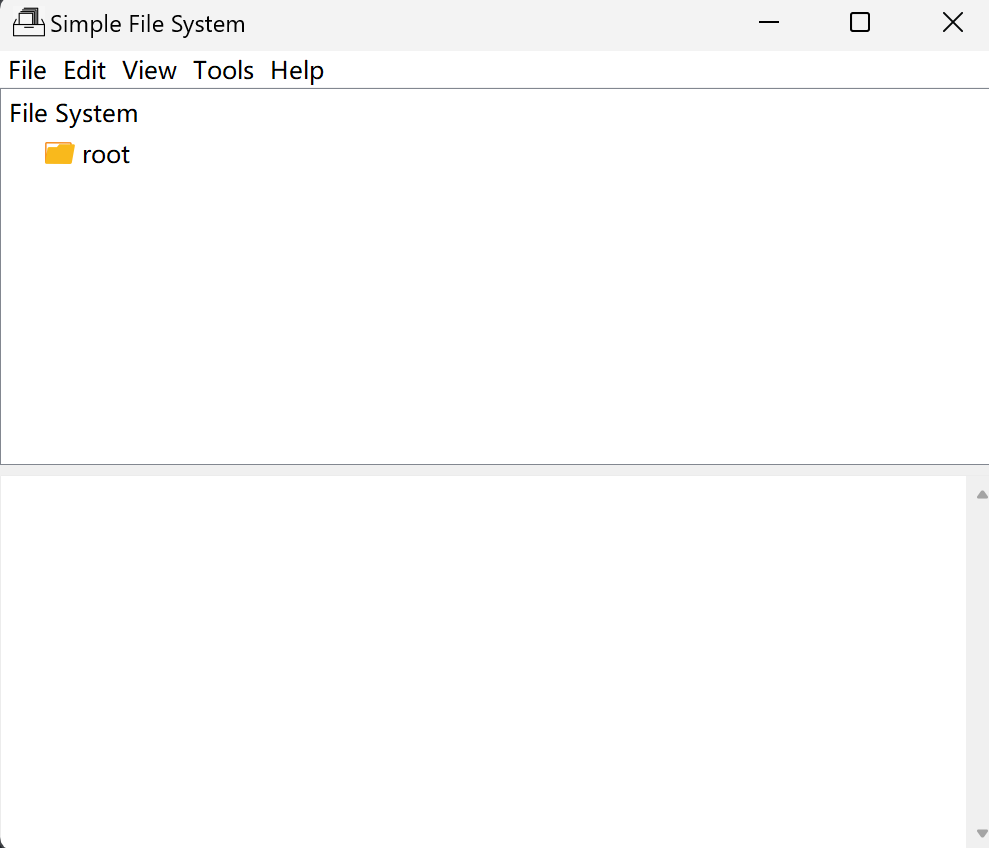
\_update\_fcb\_references: 更新文件控制块的父目录引用。

display\_message: 在文本编辑器中显示消息。

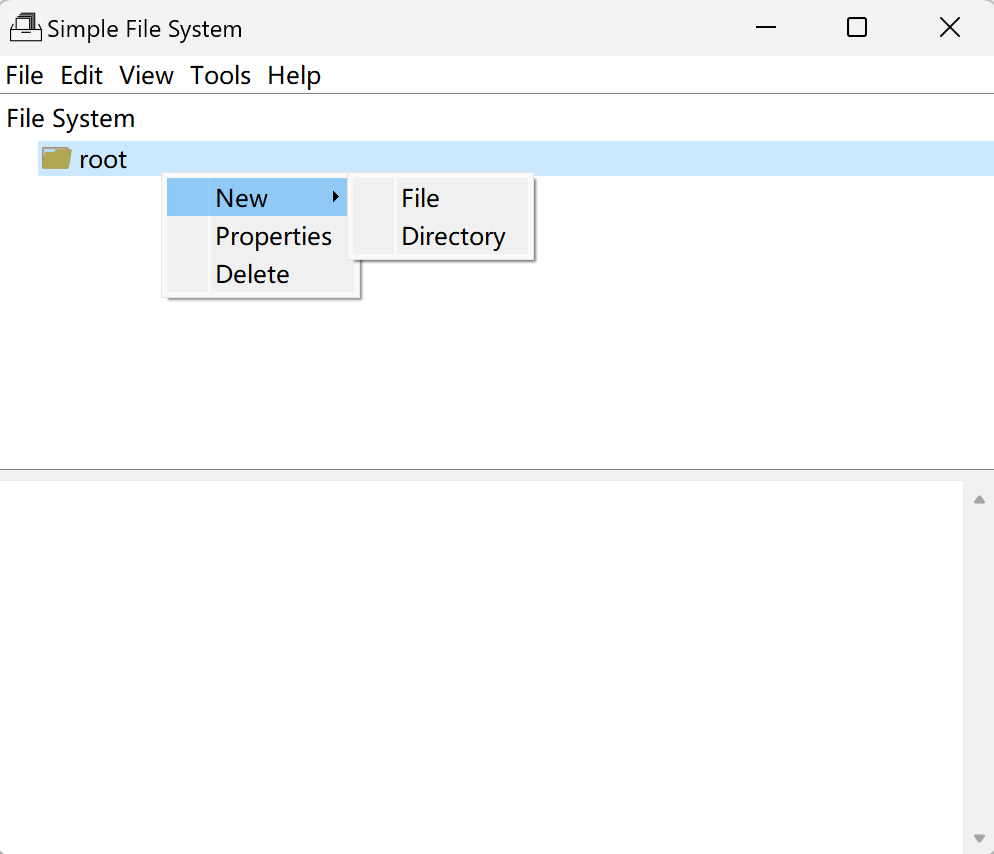
format\_disk: 确认并格式化文件系统，删除所有数据。

1. 成果展示

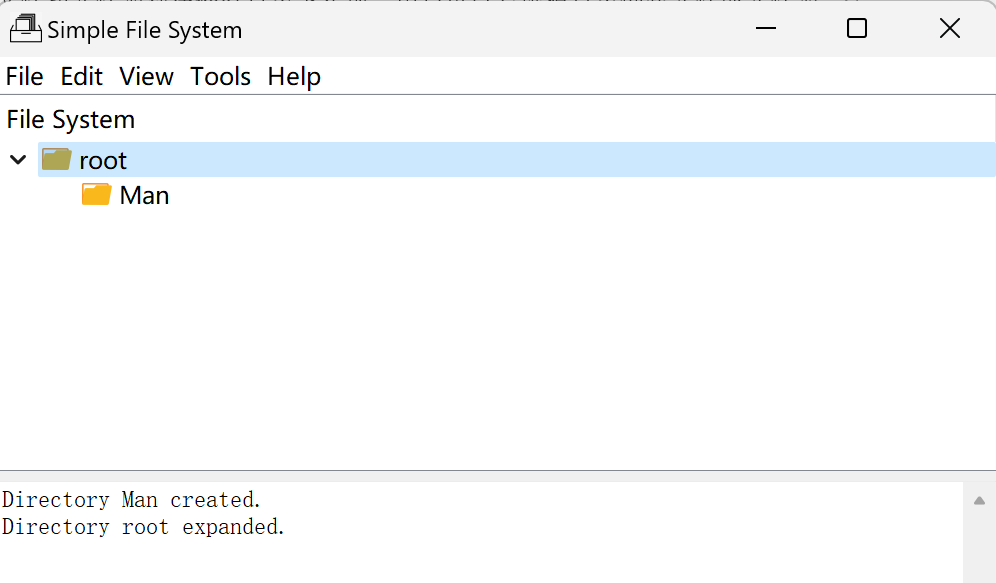
界面分为上下两部分，上面是文件系统的结构，下面是用户能看到的系统回复。界面为纯英文。



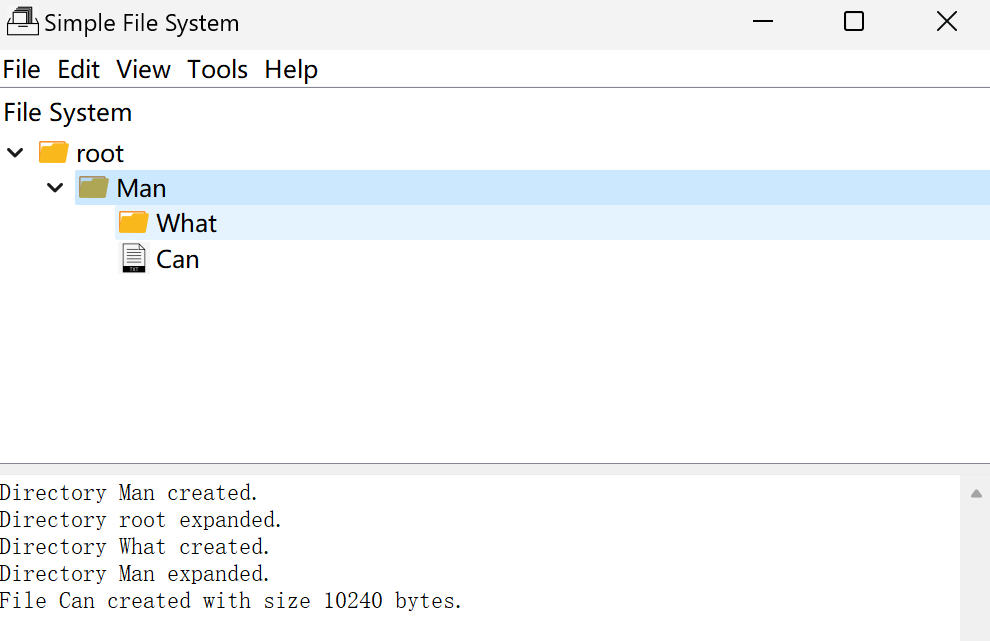
所有文件和文件夹都在root目录下实现，用户可以右键根目录创建文件或文件夹。



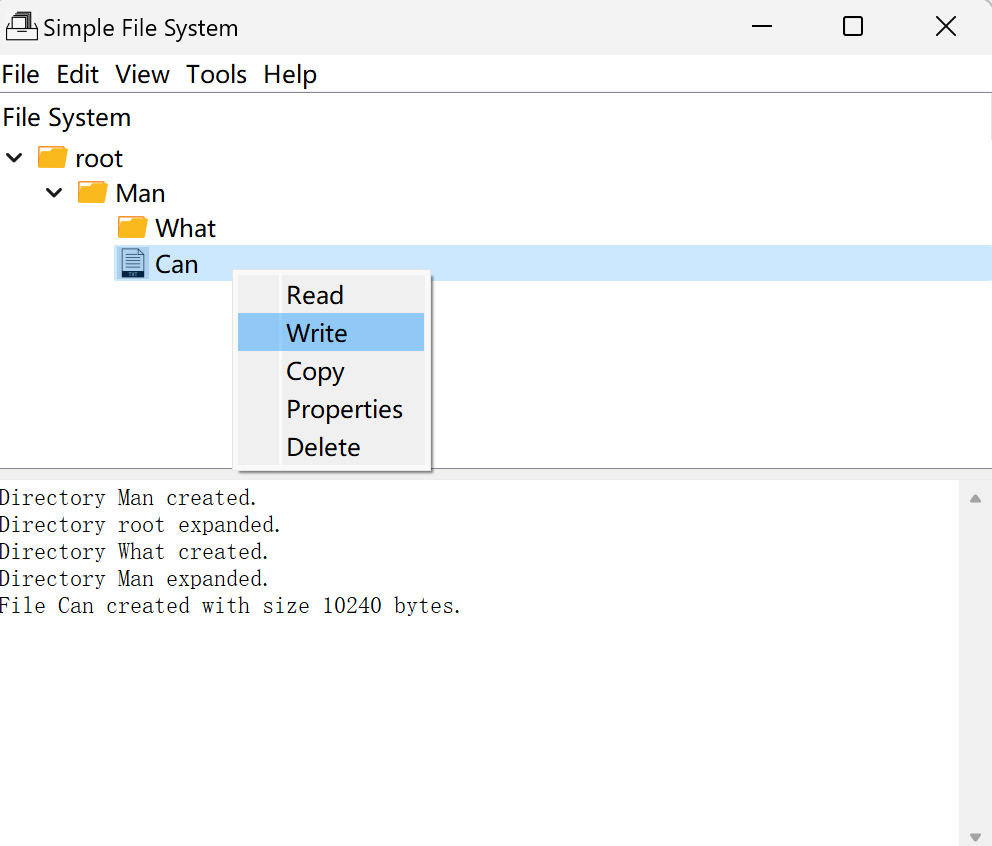
新建目录，命名为“Man”



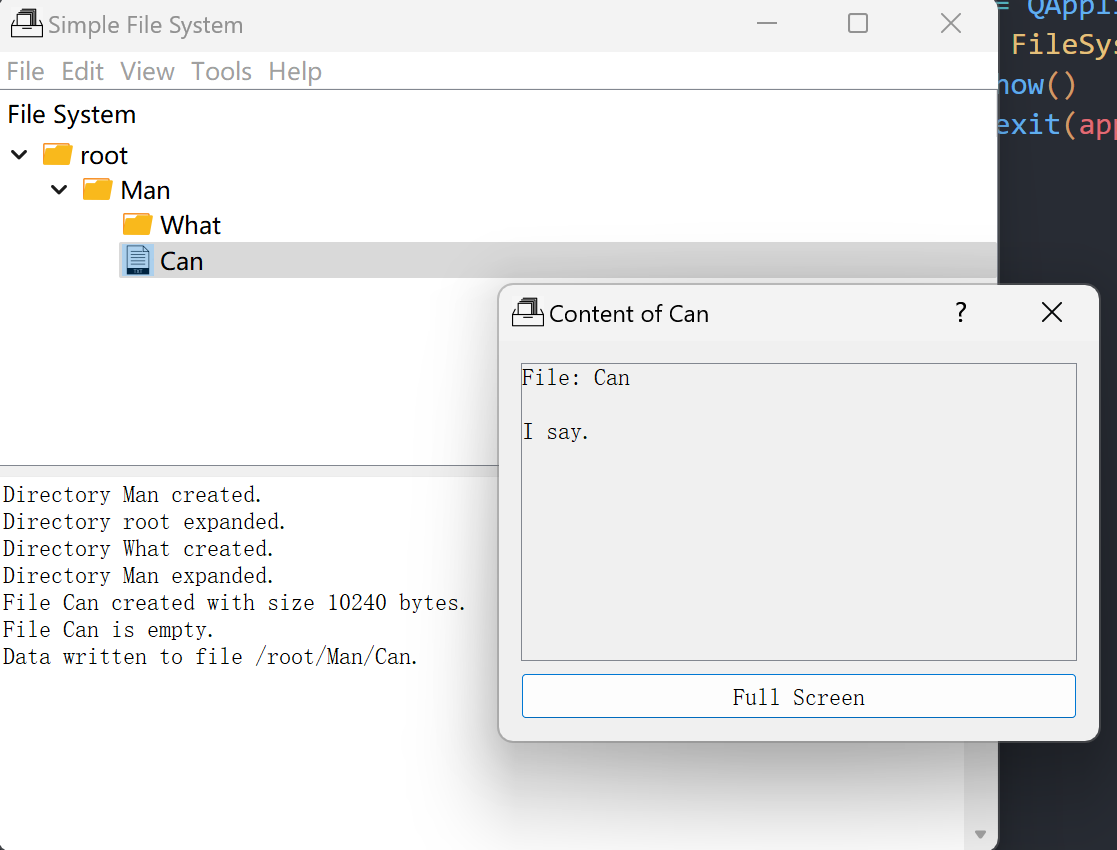
在“Man”下新建目录和文件，目录命名为“What”，文件命名为“Can”



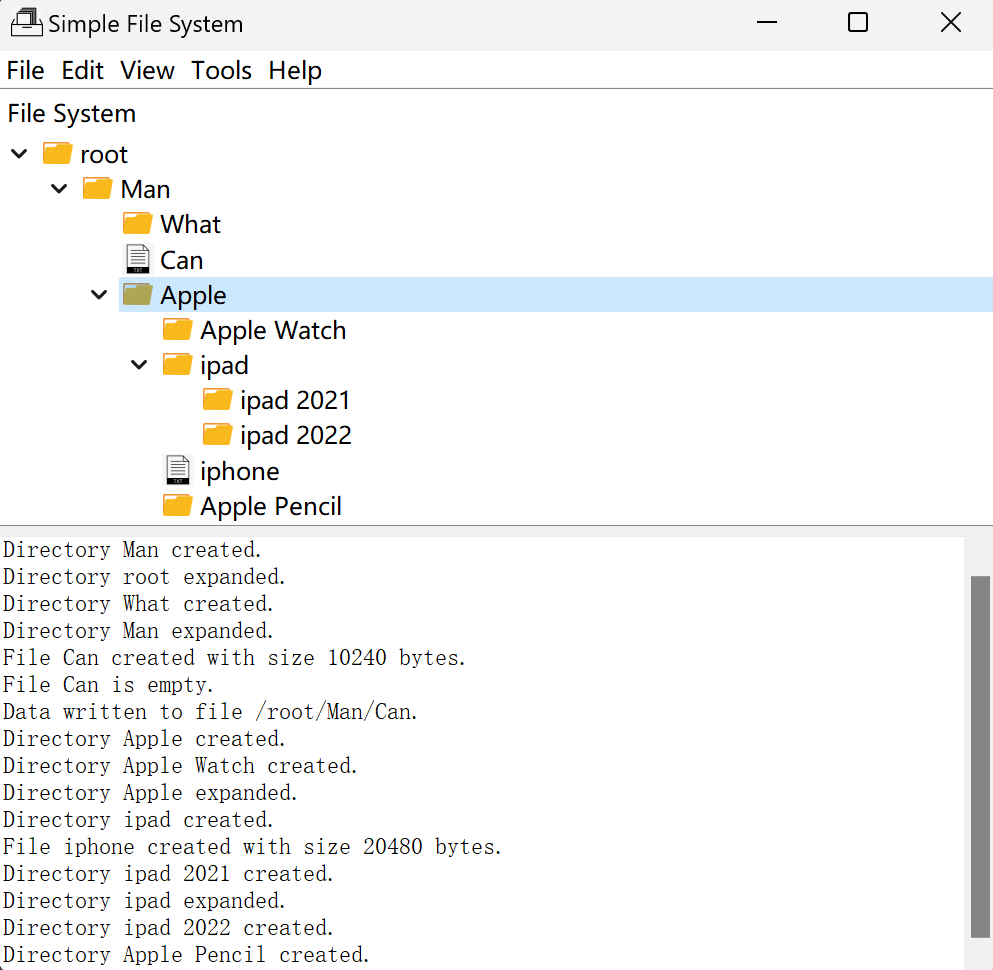
在“Can”文件内写下内容：“I say”，要求先为文件分配大小（这里使用了10KB）



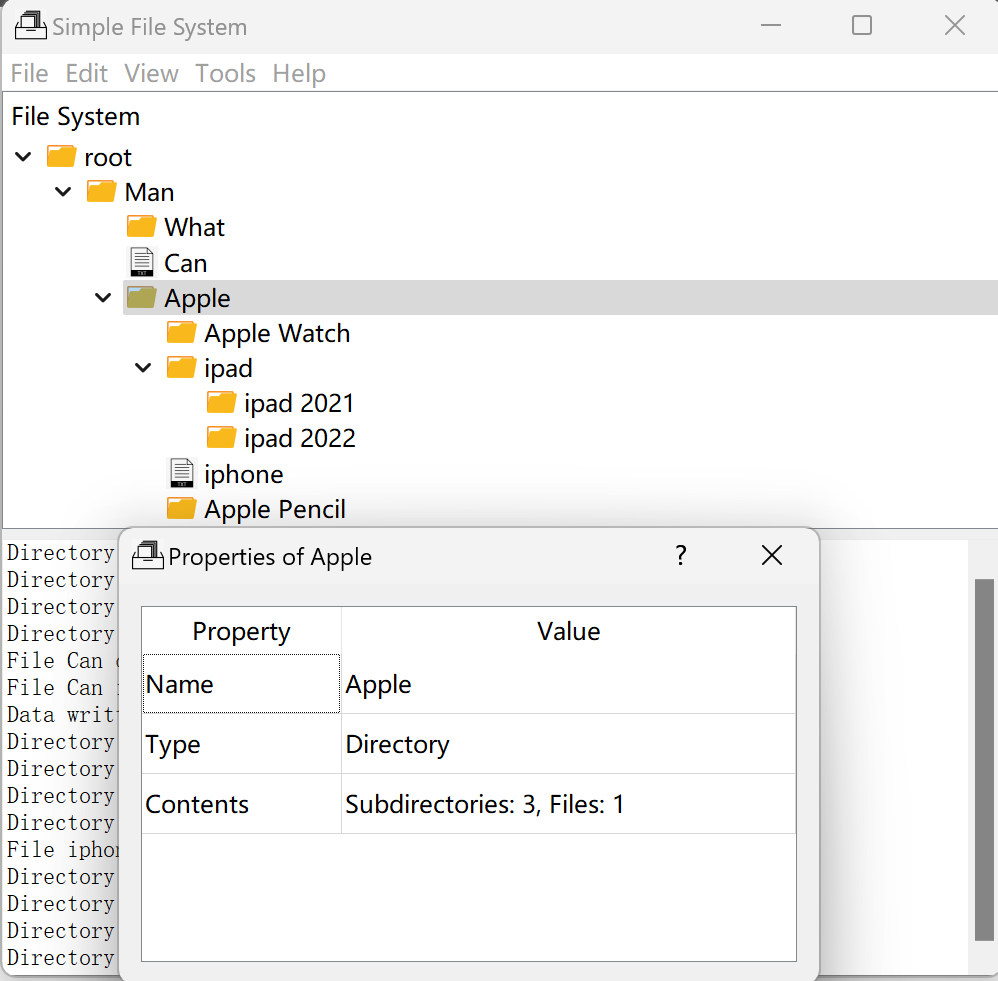
右键文件选择“read”，或者左键双击文件可以读取



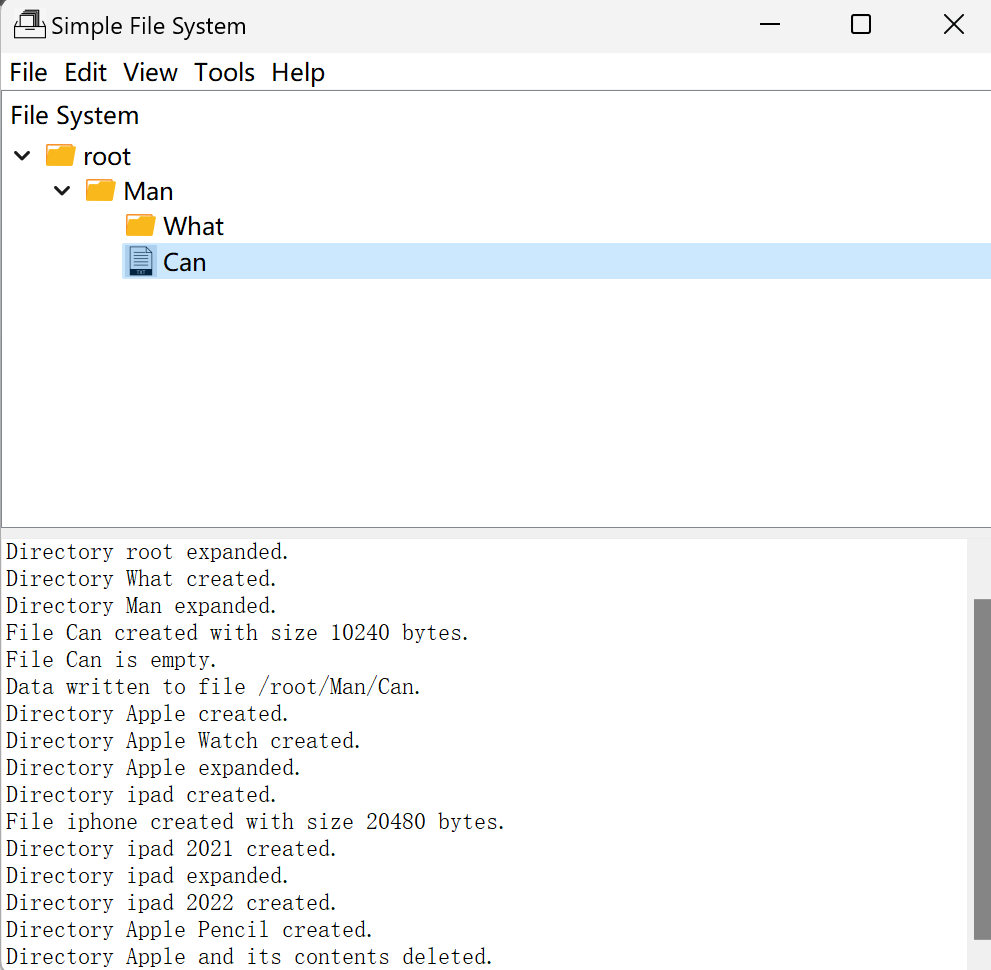
可以继续创建文件夹和文件



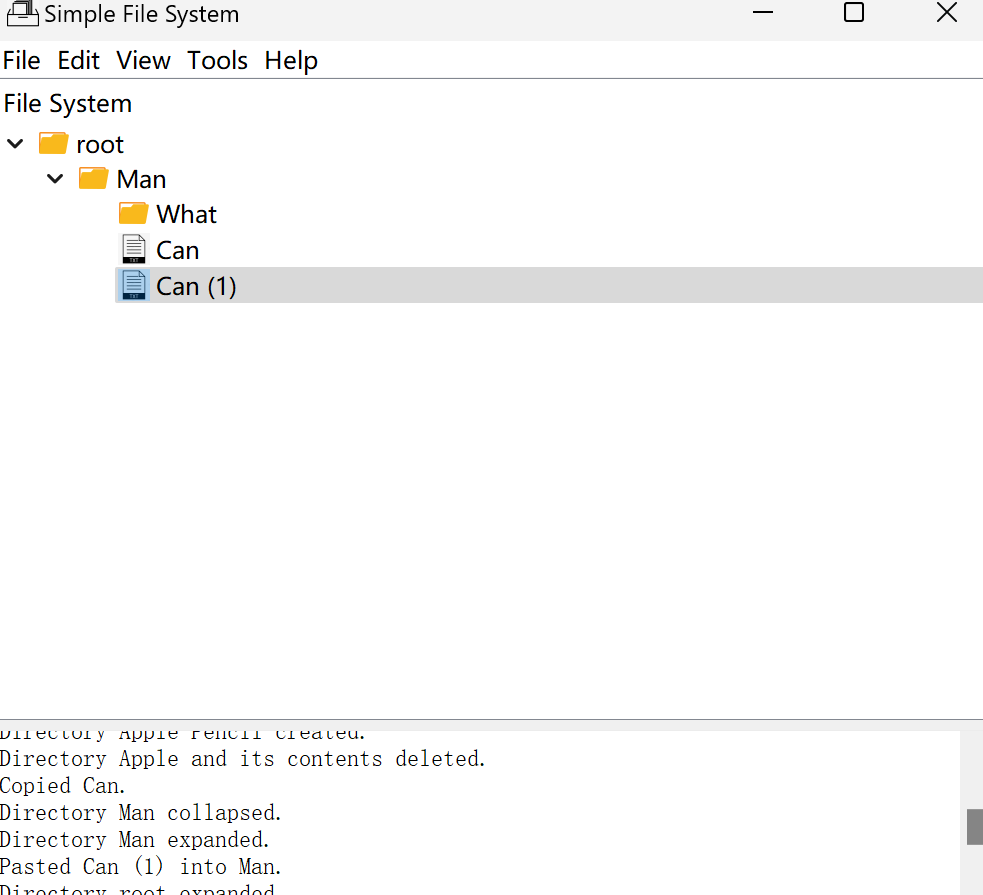
显示属性



删除文件夹“Apple”



复制粘贴文件“Can”



此外，文件系统还支持：

* + 边界检测，如判断文件大小是否超出规定大小，判断是否能开辟空间，并抛出Warning，便于维护
  + 重命名文件和文件夹
  + 保存，当退出程序或选择“Save”后，自动保存到filesystem.dat文件，下次打开时，程序先读取filesystem.dat的数据，恢复到系统
  + 刷新视图，是辅助功能，用户也可以手动刷新
  + 允许同目录下同名文件和文件夹出现，不支持同目录下同名文件（或文件夹）和文件（或文件夹）出现
  + 格式化，释放内存，系统只剩下空的根目录

更多功能，可以运行程序查看。

1. 总结
   1. 程序优点

功能方面：

1. 基本文件系统功能完备：实现了文件和目录的创建、删除、读取、写入等基本操作，满足了基础文件系统管理需求。
2. 递归处理：支持递归删除目录及其子内容，确保目录操作的完整性。
3. 文件复制和粘贴：实现了文件复制和粘贴功能，允许用户在不同目录之间移动文件。
4. 空间管理：使用FAT（文件分配表）和位图管理数据块的分配和释放，有效管理存储空间。
5. 错误处理：通过各种检查和打印错误信息提示，方便调试与维护，提高了鲁棒性。

界面方面：

* + - 1. 有目录导航：支持通过双击目录节点和右键菜单在不同目录间导航，操作方便。
      2. 界面初始化良好：在程序启动时初始化界面，包括设置窗口标题、图标、菜单栏和视图布局，用户体验良好。
      3. 视图刷新：提供视图刷新功能，确保文件系统视图与实际文件系统状态同步。
  1. 程序不足

1. 不完善的错误处理：虽然有基本的错误处理和提示，但在某些极端情况下，程序可能不能很好处理，进而导致崩溃（因为实在没有时间了就没有去花更多时间debug）。
2. 缺乏事务性操作：在执行复杂文件操作（复制、粘贴）时，没有事务机制，如果中途失败，可能导致数据不一致。
3. 说明

程序已发布到 github ，地址为 [w1ntercube/TJ-os-file\_system (github.com)](https://github.com/w1ntercube/TJ-os-file_system)