OS Project : FileManagerSystem

文件管理系统

项目分析

项目描述

在内存中开辟一块区域来模拟磁盘,做一个文件管理系统, 管理该区域的已用空间和空闲空间

开发环境

- 开发工具: VS .NET
- 开发语言:C#

需求分析

- 能够对文件进行创建, 删除, 重命名等操作
- 提供文件操作接口, 在可视化界面上进行文件打开, 文件关闭, 文件写入等操作
- 对磁盘空闲空间进行管理
- 利用多级文件目录检索文件
- 能够进行磁盘格式化
- 能够将数据可持久化, 在下次打开时恢复原状

实现方法

文件物理结构

文件采用隐式链接方式存储, 即在每个磁盘块后面都留有一段空间, 用于存储下一个磁盘块的序列号, 以达到消除外碎片, 支持文件动态增长的功能

```
for(int i = 0; i < currFCB.size; i++)
{
    string FCBName = bitToString(disk, startIndex + 2, 12);
    if (name == FCBName) return startIndex;
    //当前块已经搜完并且还有FCB存在另一个块中
    if (i >= 63)
    {
        i -= 64;
        //得到下一个该目录下的FCB
        int nextBlockNum = bitToInt(disk, startIndex + 128 - 14, 14);
        startIndex = nextBlockNum * blockSize;
    }
    else {
        startIndex += 126;
    }
}
```

```
int firstIndexToWrite = currentFCB.firstBlockNum * blockSize;
int newFolderBlock;
//当前块已满且还有下一块存有当前目录FCB
while (size > 64)
{
    size -= 64;
    int nextBlockNum = bitToInt(disk, firstIndexToWrite + blockSize - 14, 14);
    firstIndexToWrite = nextBlockNum * blockSize;
}
//当前块已满且没有下一块
if(size == 64)
{
    int nextBlockNum = getNextFreeFCBBlock();
    if (nextBlockNum == -1) return false; //无空闲FCB块装入新目录的FCB
    disk[nextBlockNum] = true;
    newFolderBlock = getNextFreeFCBBlock();
    if (newFolderBlock == -1)
    {
        disk[nextBlockNum] = false;
        return false; //无空闲FCB块装入新目录的内容
    }
    firstIndexToWrite = nextBlockNum * blockSize;
    size = 0;
}
```

空间管理

磁盘整体分为两部分: 系统区和用户区, 系统区包括位示图和FCB块, 用户区包括数据块, 本项目在设计过程中对各部分的大小进行了合理的取舍,最终结果如下:

- 磁盘每个块blocksize为1KB, 共有16*1024个块
- 位示图占据2KB, 即两个块, 对应16*1024个块
- FCB块占据256KB, 即256个块, 每个FCB大小为16B, 对应16*1024个FCB
- 数据块占据剩下的所有内存, 共有16*1024 - 258个块

```
public static int blockSize = 8 * 1024;
public static int blockNum = 16 * 1024;
public static int FCBBlockNum = 256;
public static int bitMapBlockNum = 2;
```

```
disk[0] = disk[1] = true; //前两个块存位图，标记为被占用
currentFCB = new FCB(true, true, "root", bitMapBlockNum + 1, 0);//根目录FCB赋值给当前文件
currentFCBFirstIndex = bitMapBlockNum * blockSize;
disk[bitMapBlockNum] = true; //根目录FCB在位图之后的块，标记为被占用
currentFCB.FCBWriteDisk(disk, currentFCBFirstIndex);//将根目录FCB写入对应的区域
disk[bitMapBlockNum + 1] = true;//根目录的内容存到下一个空闲的系统区块中，标记为被占用
```

文件空间管理

文件空间管理采用FCB和数据块分离的方法, 提高搜索速度的同时不会浪费空间, 下面以创建文件为例,

1. 首先查找是否有可用空间\
2. 判断当前目录对应的FCB块是否已满
3. 如果满了则需要分配一个新的FCB块, 装入该文件的FCB, 如果没满则直接创建FCB并填入
4. 为该文件分配数据块
5. 将数据写入数据块,如果一个块装不下需要向后延伸
6. 如果空间不足以装下整个数据, 则从中截断并返回

```
1 个引用
public bool writeFile(string content)
{
    if (currentFCB.type) return false;
    clearFileBlock();

    int contentByte = content.Length;//剩余未写入磁盘的字节数
    const int blockFilebyte = 1021;//一个数据块最多能写入的字节数
    int stringstart = 0;//下一次要写入的数据在原字符串位置
    int blockIndexToWrite = currentFCB.firstBlockNum * blockSize;
    disk[currentFCB.firstBlockNum] = true;

    while (contentByte > 0)
    {

        int blockLength = contentByte > blockFilebyte ? blockFilebyte : contentByte; //要写入该数据块的字节数

        IntToBit(disk, blockIndexToWrite, blockLength, 10);
        stringToBit(disk, blockIndexToWrite + 10, stringstart, blockLength, content);

        contentByte -= 1021;
        stringstart += 1021;

        currentFCB.size++;
        currentFCB.FCBWriteDisk(disk, currentFCBFirstIndex);

        if(contentByte > 0)
        {
            int nextBlockNum = getNextFreeFileBlock();
            if (nextBlockNum == -1) return false;//可用空间不足
            IntToBit(disk, blockIndexToWrite + blockSize - 14, nextBlockNum, 14);
            blockIndexToWrite = nextBlockNum * blockSize;
            disk[nextBlockNum] = true;
        }
    }
    return true;
}
```

```
1 个引用
public bool createFile(string fileName)
{
    if(fileName.Length > 12 || fileName.Length == 0|| !currentFCB.type) return fals
    int findFCBFirstIndex = findFCB(currentFCBFirstIndex, fileName);
    if (findFCBFirstIndex != -1) return false; //出现重名，拒绝创建请求
    int size = currentFCB.size;
    int firstIndexToWrite = currentFCB.firstBlockNum * blockSize;
    int newFileBlock;
    //当前块已满且还有下一块存有当前目录FCB
    while (size > 64)
    {
        size -= 64;
        int nextBlockNum = bitToInt(disk, firstIndexToWrite + blockSize - 14, 14);
        firstIndexToWrite = nextBlockNum * blockSize;
    }
    //当前块已满且没有下一块
    if (size == 64)
    {
        int nextBlockNum = getNextFreeFCBBlock();
        if (nextBlockNum == -1) return false; //无空闲FCB块装入新文件的FCB
        disk[nextBlockNum] = true;
        newFileBlock = getNextFreeFileBlock();
        if (newFileBlock == -1)
        {
            disk[nextBlockNum] = false;
            return false; //无空闲FCB块装入新文件的内容
        }
        firstIndexToWrite = nextBlockNum * blockSize;
        size = 0;
    }
    firstIndexToWrite += size * 126;
    newFileBlock = getNextFreeFileBlock();
    if (newFileBlock == -1) return false; //无空闲FCB块装入新目录的内容
    disk[newFileBlock] = true;
    FCB newFileFCB = new FCB(true, false, fileName, newFileBlock, 0);
    newFileFCB.FCBWriteDisk(disk, firstIndexToWrite);
    currentFCB.size++;
    currentFCB.FCBWriteDisk(disk, currentFCBFirstIndex);
    return true;
}
2 个引用
```

空闲空间管理

空闲空间管理采用位示图方法, 当需要寻找一个空的磁盘块时, 会遍历位示图得到最前面的块,当释放某文件或目录时,也会归还空闲空间, 提高利用效率

```
public void delete(int FCBFirstIndex)
{
    FCB FCBToDelete = new FCB(disk, FCBFirstIndex);
    if(FCBToDelete.type == true)
    {
        //如果是目录，需要递归删除目录中所有子目录和文件
        int subFCBFirstIndex = FCBToDelete.firstBlockNum * blockSize;
        for (int i = 0; i < FCBToDelete.size; i++)
        {
            delete(subFCBFirstIndex);
            subFCBFirstIndex += 126;
            if(((i + 1) % 64) == 0)
            {
                int nextBlockNum = bitToInt(disk, subFCBFirstIndex + 128 - 14, 14);
                subFCBFirstIndex = nextBlockNum * blockSize;
            }
        }
        //对FCB宣称占有的空间进行清除
        int size = FCBToDelete.size;
        int subFCBBlockNum = FCBToDelete.firstBlockNum;
        while (size > 0)
        {
            //先记录下一个块的序号
            int nextBlockNum = bitToInt(disk, (subFCBBlockNum + 1) * blockSize - 14, 14);
            setDiskInit(subFCBBlockNum * blockSize, blockSize);
            disk[subFCBBlockNum] = false; //维护位示图
            subFCBBlockNum = nextBlockNum;
            size -= 64;
        }

        setDiskInit(FCBFirstIndex + 112, 14);//将父目录的size置为0
    }
```

目录结构

目录采用多级目录结构, 高级目录的FCB所指向的内容还是FCB, 从而可以创建同名文件, 也为多用户提供了基础.

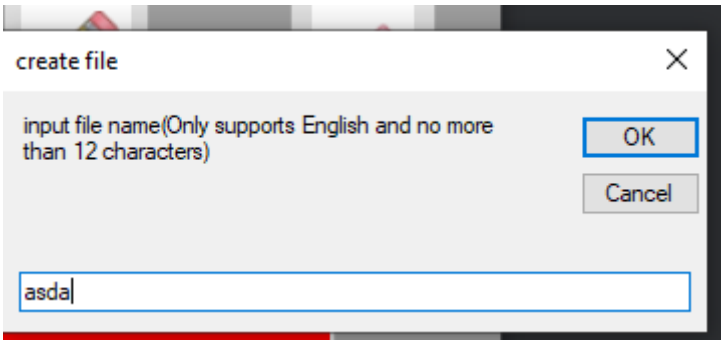
```
6 个引用
public int findFCB(int FCBStart, string name)
{
    if(disk[FCBStart] == false || disk[FCBStart + 1] == false)
        throw new System.InvalidOperationException("can't find a non existent floder!");
    FCB currFCB = new FCB(disk, FCBStart);
    int startIndex = currFCB.firstBlockNum * blockSize;
    for(int i = 0; i < currFCB.size; i++)
    {
        string FCBName = bitToString(disk, startIndex + 2, 12);
        if (name == FCBName) return startIndex;
        //当前块已经搜完并且还有FCB存在另一个块中
        if (i >= 63)
        {
            i -= 64;
            //得到下一个该目录下的FCB
            int nextBlockNum = bitToInt(disk, startIndex + 128 - 14, 14);
            startIndex = nextBlockNum * blockSize;
        }
        else {
            startIndex += 126;
        }
    }
    //如果找不到 则返回-1
    return -1;
}
```

成果展示

初始界面,

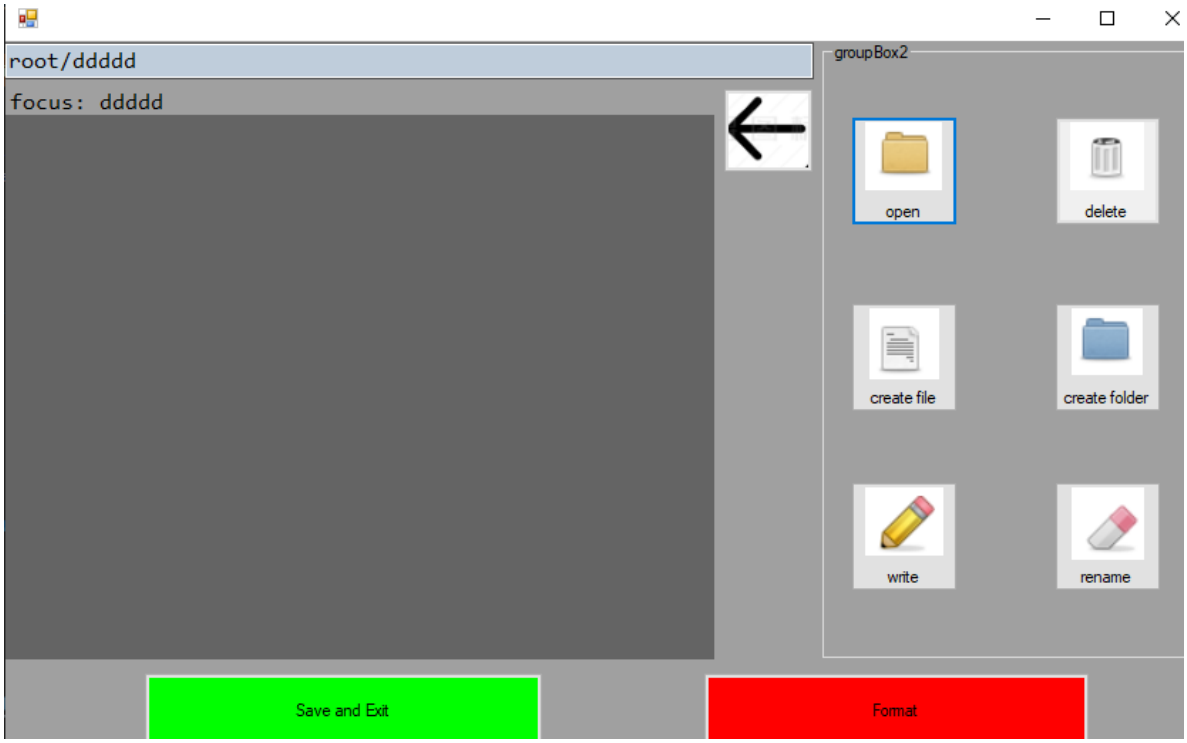

root
ready!
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

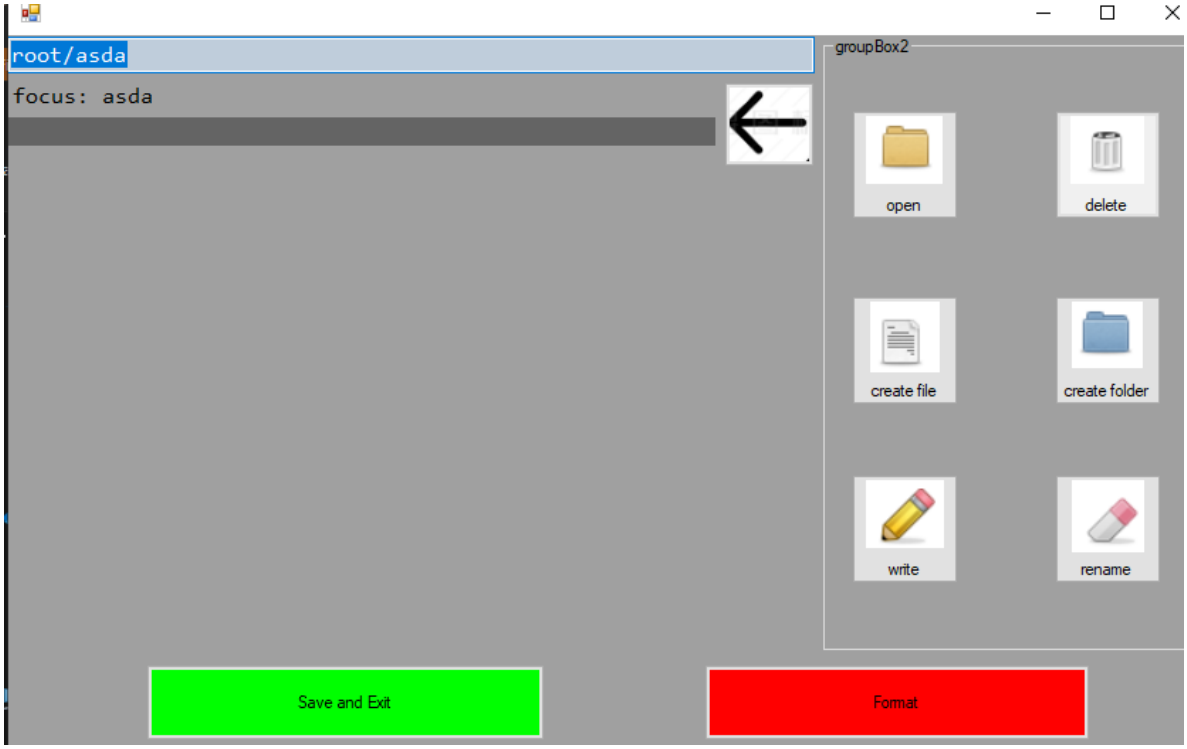
创建


create file
input file name(Only supports English and no more than 12 characters)
OK
Cancel
asda

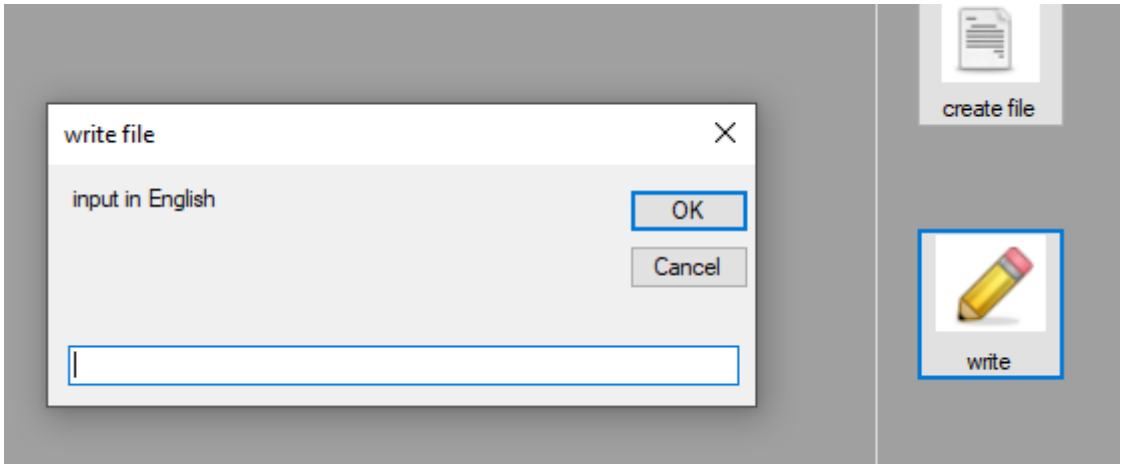
打开文件/文件夹

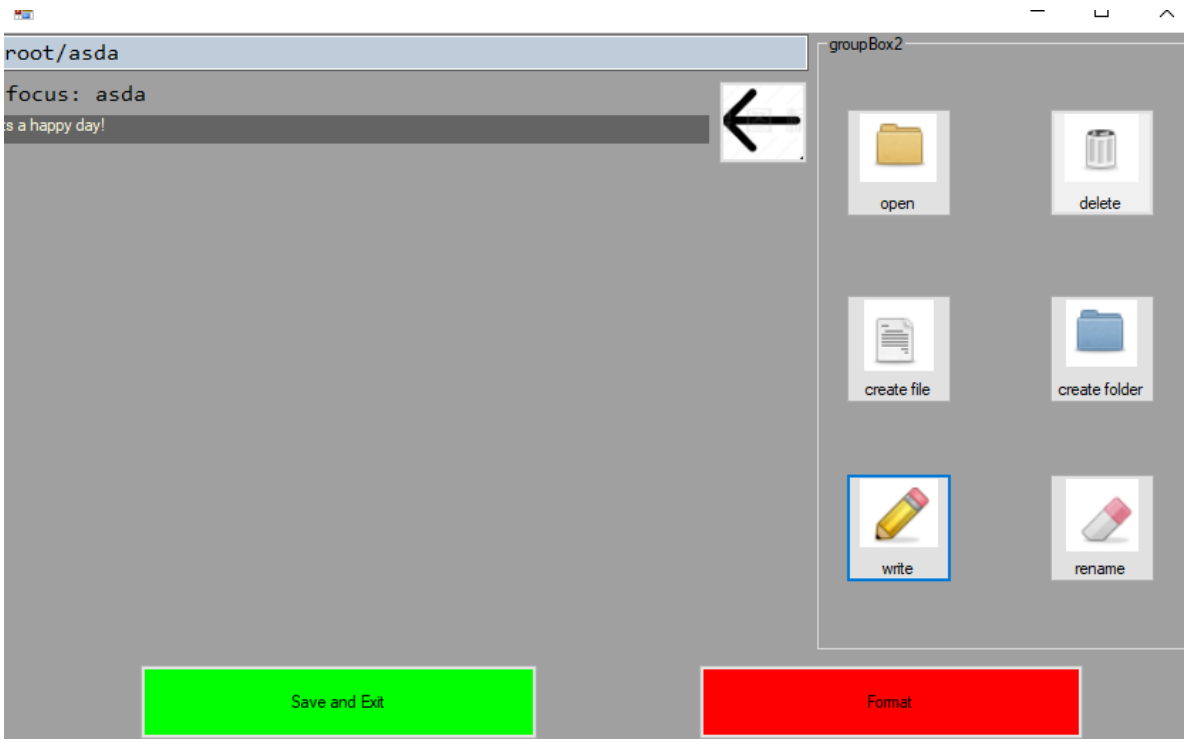

root
focus: ddddd
file
asda
folder
ddddd
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format


root/ddddd
focus: ddddd
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

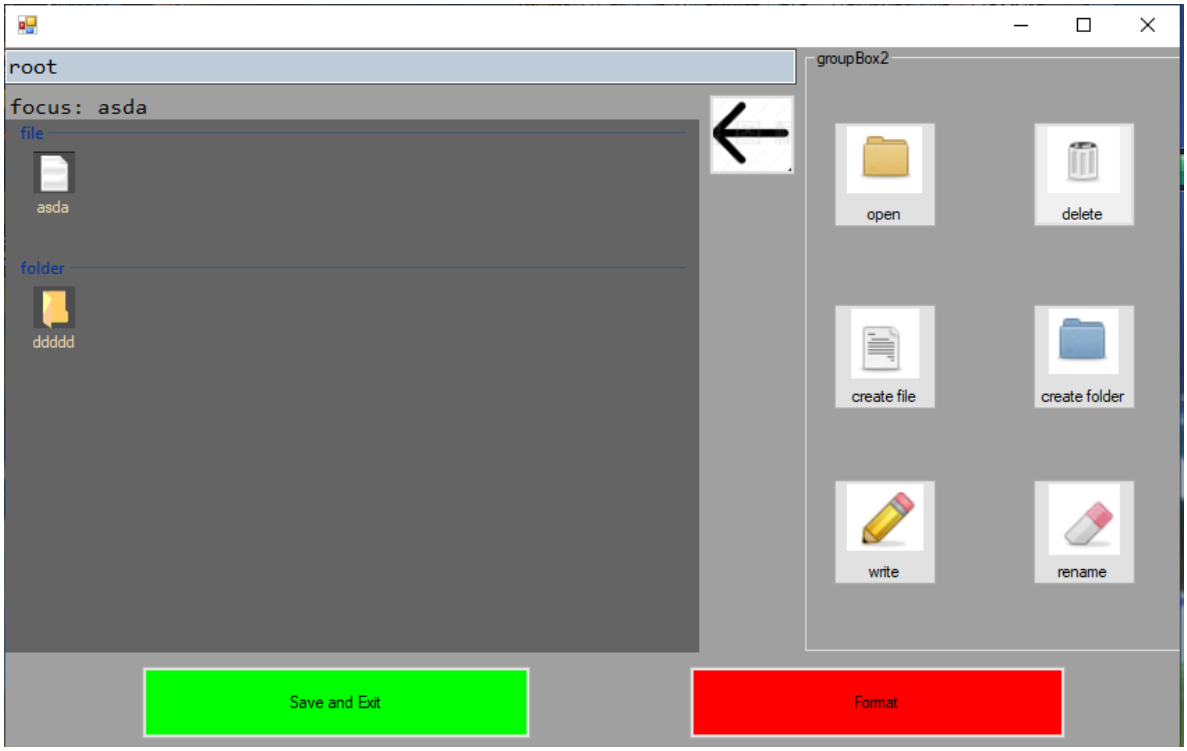

root/asda
focus: asda
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

写入文件

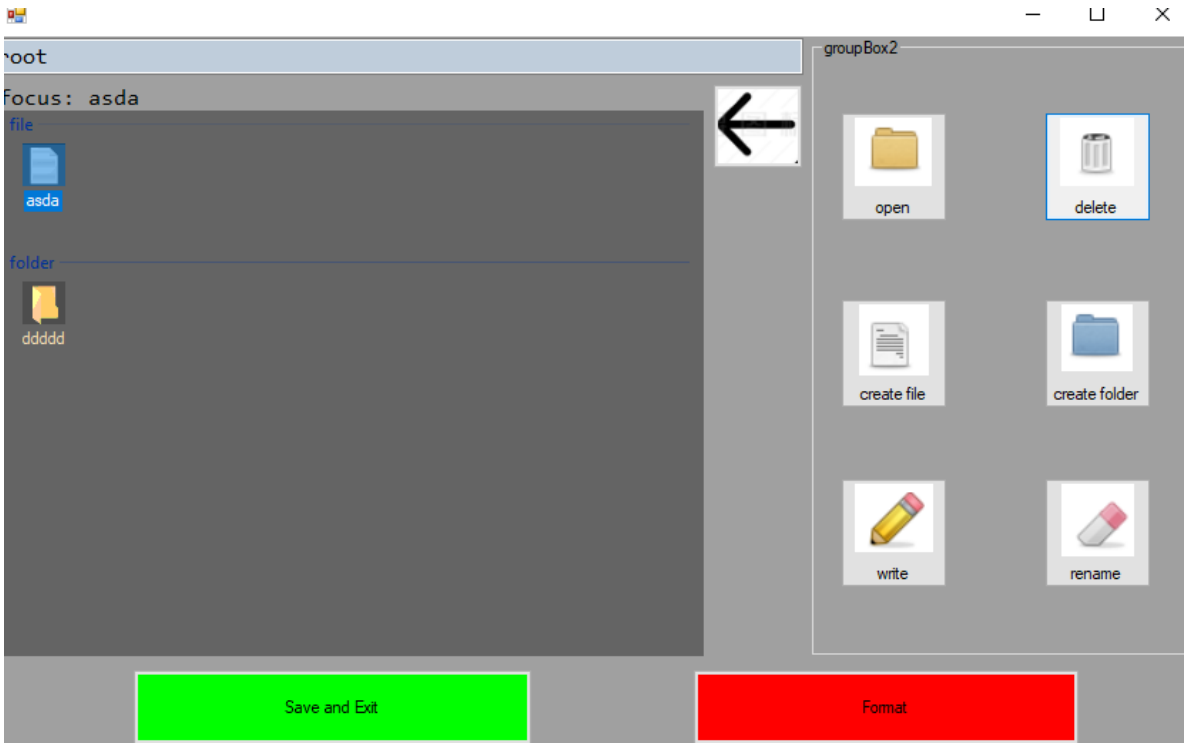

write file
input in English
OK
Cancel
create file
write

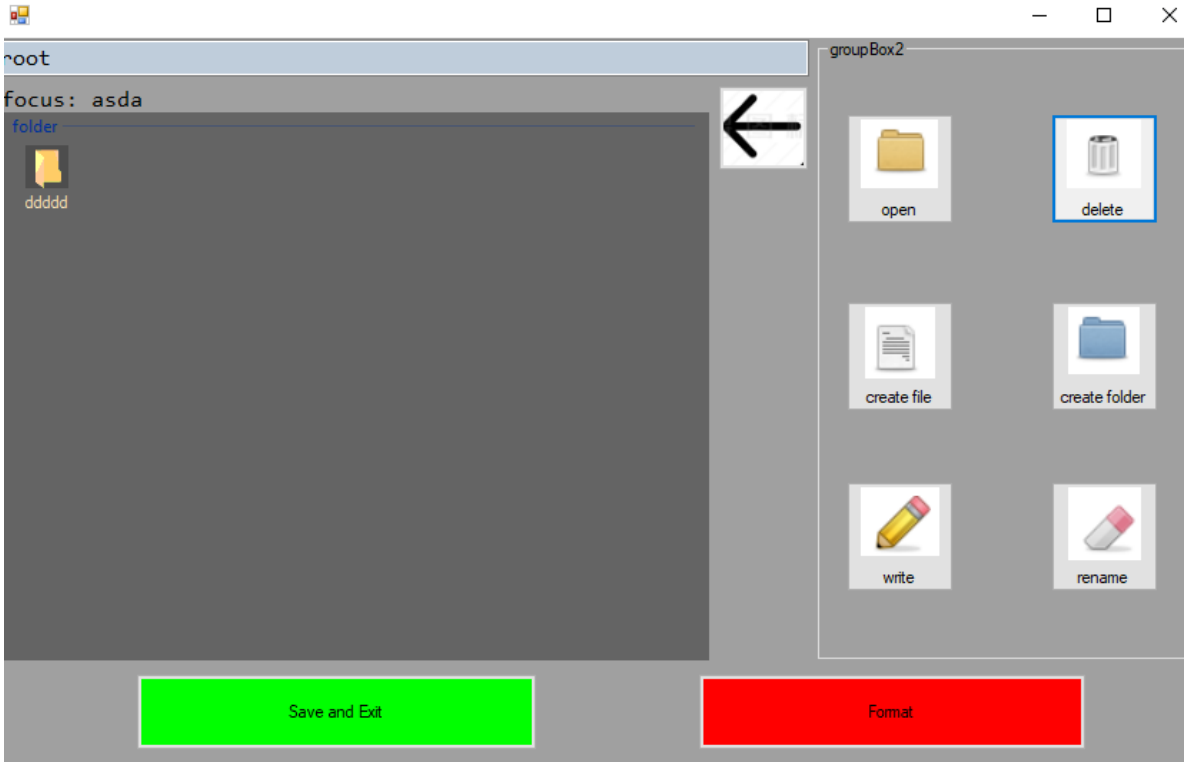

root/asda
focus: asda
s a happy day!
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

回退

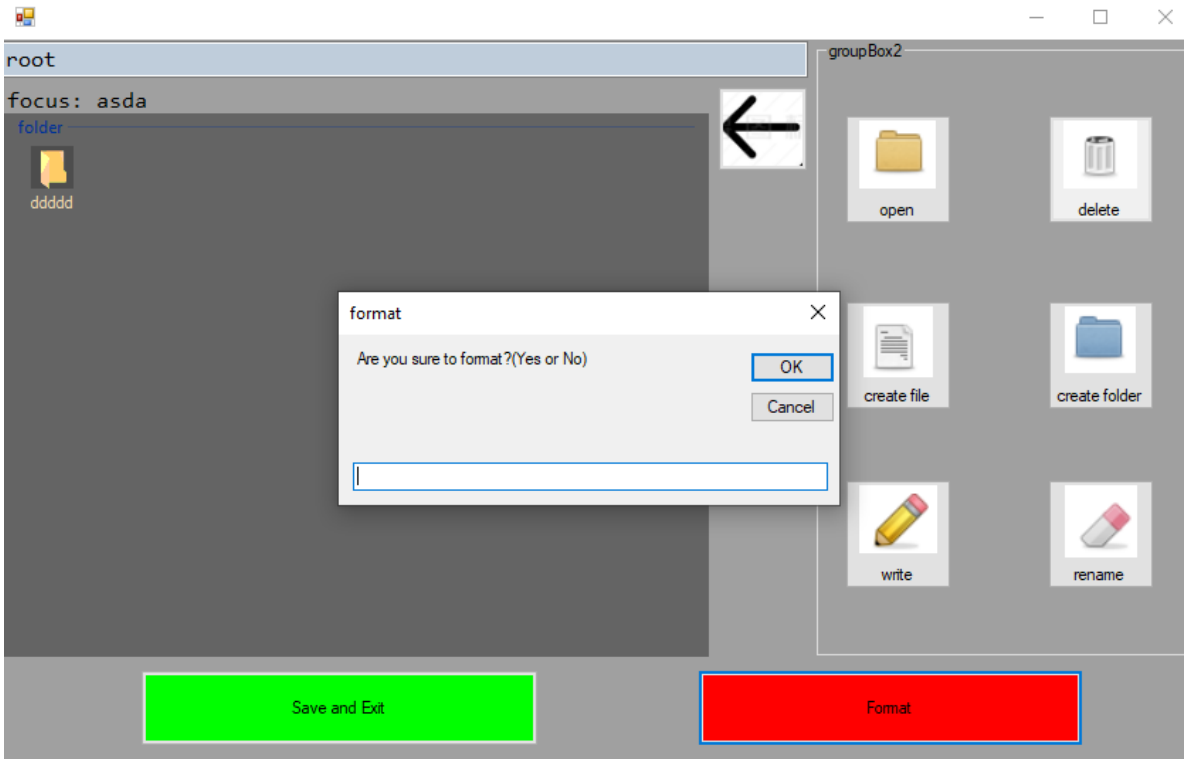

root
focus: asda
file
asda
folder
ddddd
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

删除文件


root
focus: asda
file
asda
folder
ddddd
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format


root
focus: asda
folder
ddddd
groupBox2
open
delete
create file
create folder
write
rename
Save and Exit
Format

格式化


root
focus: asda
folder
ddddd
groupBox2
open
delete
create file
create folder
write
rename
format
Are you sure to format?(Yes or No)
OK
Cancel
Save and Exit
Format