**Jugg文件系统程序设计文档**

**一、概述**

Jugg文件系统实现方式为

1，VC程序创建一个大的（如2G）.jugg文件作为一个虚拟分区

2，在该虚拟分区上使用类似于linux ext2的文件系统格式，.exe程序是文件上传、下载、删除的接口程序。

实现的功能：

1，查看文件系统中现有文件列表、创建日期、文件大小等，并可已不同属性进行排序，类似于Windows资源管理器。

2，用户可以直接拖拽文件实现上传，并且可以同时拖拽多个文件。

3，可以多选择的删除、下载文件，删除还能对文件夹进行操作。

4，实现文件夹的存放，双击进入文件夹

5，可以移动文件系统中的文件夹和文件，并且可以直接由拖动实现。

**二、整体设计**

文件系统通常会将这两部分的数据分别存放在不同的块，权限与属性放置到node中，至于实际数据则放置到data bloke块中。另外，还有一个超级块会记录整个文件系统的整体信息，包括node与block的总量、使用量、剩余量等。由于每个node与block块都有编号，而每个文件都会占用一个node，node内则有文件数据放置文件所放置数据的block号码，就能够读出该文件的实际数据了。

node的内容用于记录文件的权限与相关属性，至于block块则是在记录文件的实际内容，而且文件的一开始就将node与block规划好了，除非重新格式化，否则node与block固定后就不再变动。

Jugg文件系统的参数细节：



1，整个分区结构如上图所示，程序中默认分区大小为2G。

2，其中JuggRule固定占用1KB，中间记录了Node和Block的大小、数量、其它段的起始地址。本程序中Node大小为256B，支持最大文件/文件夹名称长度为160个字节，长度为15的int索引数组；Block大小为2KB。

3，文件系统为32位的文件系统，支持最大分区为 2G×2KB=4TB。

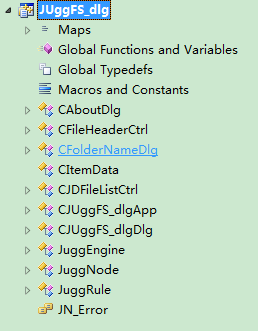
4，索引分配使用混合索引分配方式，可参考下图ext2的索引方式。最大支持的单个文件大小为 512×512×512×2KB=256GB。（512为单个block可以存放的block地址数量）

5，本文件系统的目录结构为多级目录结构，即一般性的树型结构。在文件、程序中通过孩子-兄弟二叉链表来存储表示。

**三、实现层次**

1，主要的类

|  |  |  |
| --- | --- | --- |
| 层次 |  |  |
| 界面层 | 对话框类、列表控件（列表头）类 |  |
| 文件系统引擎层 | JuggEngine |  |
| 底层数据层 | JuggRule JuggNode JN\_Error（枚举定义） |  |



2，关键算法

1)、二叉树的遍历

二叉树的遍历在搜索和删除二叉树的时候都需要用到。本程序中二叉树的遍历使用无递归的中根顺序，其中删除二叉树代码如下

stack<JuggNode\*> del\_stack;

JuggNode \*pJN,\*pTmp;

pJN=m\_pRootNode;

while(pJN || !del\_stack.empty()){

if(pJN){

del\_stack.push(pJN);

pJN=pJN->m\_pLChild;

}

else{

pJN=del\_stack.top();

del\_stack.pop();

pTmp=pJN->m\_pRChild;

delete pJN;

pJN=pTmp;

}

}

m\_pRootNode=NULL;

2)、三级索引列表中文件内容的读取

下列代码为使用三级索引时的代码

int tmp,main\_item\_cnt;//;

tmp=bcks\_num%TwoStage;

main\_item\_cnt=(bcks\_num-tmp)/TwoStage;//主block数量

if(tmp){

main\_item\_cnt++;

}

readed\_cnt=0;

DownloadStageBlockBuf(bbuf\_main,pNode->jn\_block[0]-1);//载入主block

for(int i=0;i<main\_item\_cnt;i++){//遍历主block

DownloadStageBlockBuf(bbuf\_secnd,bbuf\_main[i]-1);//载入次block

for(int j=0;j<OneStage;j++){//遍历整个次block

DownloadStageBlockBuf(bbuf\_third,bbuf\_secnd[j]-1);//载入第三block

for(int k=0;k<OneStage;k++){//遍历第三block

if(readed\_cnt==bcks\_num-1){

DownloadBlockByte(dFile,bbuf\_third[k]-1,d512\_rmn);//读取数据

return JNE\_Correct;//结束标志

}

else{

DownloadBlockByte(dFile,bbuf\_third[k]-1);//读取数据

readed\_cnt++;

}

}//end 第三

}//end次

}//end 主