c++ 查找文件夹下所有文件

(2011-09-20 15:01:24)

[[http://simg.sinajs.cn/blog7style/images/common/sg_trans.gif](javascript:;)转载▼](javascript:;)

|  |  |
| --- | --- |
| 标签：  [杂谈](http://search.sina.com.cn/?c=blog&q=%D4%D3%CC%B8&by=tag) |  |

原文：http://social.msdn.microsoft.com/Forums/zh-CN/visualcpluszhchs/thread/e31f4dd3-a3c6-4b79-ae2b-e26cfd736cf6

参考：http://msdn.microsoft.com/zh-cn/library/3k3z0k7a(v=VS.90)

=========================================================================

要获取在一个文件夹中的所有文件的名字，首先，我们必须调用[FindFirstFile](http://msdn.microsoft.com/en-us/library/aa364418(VS.85).aspx)函数来打开一个搜索句柄和获得在文件夹中的文件系统中的第一个文件的信息。之后，我们必须调用[FindNextFile](http://msdn.microsoft.com/en-us/library/aa364428(VS.85).aspx)函数在调用FindFirstFile之后列出接下来所有的文件。

例子：

WIN32\_FIND\_DATA FindFileData;

HANDLE hFind;

TCHAR \*FilePathBuff = L"C:\\TestFolder";

hFind = FindFirstFile(FilePathBuff, &FindFileData);

if (hFind == INVALID\_HANDLE\_VALUE)

{

\_tprintf(TEXT("FindFirstFile failed (%d)\n"), GetLastError());

}

else

{

\_tprintf(TEXT("The first file is %s\n"), FindFileData.cFileName);

while (FindNextFile(hFind, &FindFileData) != 0)

{

\_tprintf(TEXT("The next file is %s\n"),

FindFileData.cFileName);

}

FindClose(hFind);

## C++怎样搜索文件

[检举](http://zhidao.baidu.com/question/110569459.html)|2009-08-07 09:18 提问者： [yykkyywill](http://www.baidu.com/p/yykkyywill?from=zhidao) |浏览次数：2118次

在文件夹中有1.txt-99.txt，一共99个文件，我输入55，就会打开55.txt,同时复制到b.txt

我来帮他解答

图片



符号

编号

排版

地图

[检举](http://zhidao.baidu.com/question/110569459.html)|2009-08-07 12:21

#include<iostream>  
#include<fstream>  
using namespace std;  
  
int main()  
{  
 char fileName[20],suffix[]=".txt";  
 cin>>fileName;  
 strcat(fileName,suffix);  
 ifstream in(fileName,ios::binary);  
 ofstream out("b.txt",ios::binary);  
 char temp;  
 while(in.peek()!=-1)  
 {  
 in.read(&temp,sizeof(temp));  
 out<<temp;  
 }  
 system(fileName);  
 return 0;  
}

[**C++文件搜索**](http://blog.csdn.net/fjclc2008/article/details/5739142)

2010-07-16 11:06162人阅读[评论](http://blog.csdn.net/fjclc2008/article/details/5739142#comments)(0)[收藏](javascript:void(0);)[举报](http://blog.csdn.net/fjclc2008/article/details/5739142#report)

//支持子目录，隐藏文件，只读文件的查找   
//使用方法：用命令行输入。例如，程序名为Find.exe，则命令行为 Find yourfile   
//查找支持通配符\*,?

//支持命令行

#define \_WIN32\_WINNT 0x0400

#include <iostream>   
#include <stdlib.h>   
using namespace std;

#ifdef UNICODE   
#undef UNICODE   
#endif

#include <windows.h>

BOOL MyFindFile(LPCSTR sFindPath, LPCSTR sFindFileName, ULONGLONG &uCountFolder, ULONGLONG &uCountFile)   
{   
char sPath[MAX\_PATH];   
char sFormatFileName[MAX\_PATH+2] = "\*";   
WIN32\_FIND\_DATA FindFileData;   
HANDLE hFind;   
BOOL fFinished = FALSE;

lstrcpy(sFormatFileName, sFindPath);   
lstrcat(sFormatFileName, "//\*");   
lstrcat(sFormatFileName, sFindFileName);   
lstrcat(sFormatFileName, "\*");

hFind = FindFirstFile(sFormatFileName, &FindFileData);

if (hFind == INVALID\_HANDLE\_VALUE)   
{   
return FALSE;   
}   
else   
{   
while (!fFinished)   
{   
lstrcpy(sPath, sFindPath);   
lstrcat(sPath, "//");   
lstrcat(sPath, FindFileData.cFileName);

if (FILE\_ATTRIBUTE\_DIRECTORY & FindFileData.dwFileAttributes)   
{   
if (0 != lstrcmp(FindFileData.cFileName,".") && 0 != lstrcmp(FindFileData.cFileName,".."))   
cout << " Folder " << ++uCountFolder << ". - " << sPath <<""<< endl;   
}   
else   
cout << " File " << ++uCountFile << ". - " << sPath <<""<< endl;

if (!FindNextFile(hFind, &FindFileData))   
{   
if (GetLastError() == ERROR\_NO\_MORE\_FILES)   
{   
fFinished = TRUE;   
}   
else   
{   
break;   
}   
}   
}

FindClose(hFind);   
}

return TRUE;   
}

BOOL MyFindFolder(LPCSTR sPath, LPCSTR sFindFileName, ULONGLONG &uCountFolder, ULONGLONG &uCountFile)   
{   
char sTemp[MAX\_PATH];   
char sFormatFileName[MAX\_PATH];   
WIN32\_FIND\_DATA FindFileData;   
HANDLE hFind;   
BOOL fFinished = FALSE;

MyFindFile(sPath, sFindFileName, uCountFolder, uCountFile);

lstrcpy(sFormatFileName, sPath);   
lstrcat(sFormatFileName, "//\*");   
hFind = FindFirstFile(sFormatFileName, &FindFileData);

if (hFind == INVALID\_HANDLE\_VALUE)   
{   
return FALSE;   
}   
else   
{   
while (!fFinished)   
{   
if (FILE\_ATTRIBUTE\_DIRECTORY & FindFileData.dwFileAttributes)   
{   
if (0 != lstrcmp(FindFileData.cFileName,".") && 0 != lstrcmp(FindFileData.cFileName,".."))   
{   
lstrcpy(sTemp, sPath);   
lstrcat(sTemp, "//");   
lstrcat(sTemp, FindFileData.cFileName);   
MyFindFolder(sTemp, sFindFileName, uCountFolder, uCountFile);   
}   
}

if (!FindNextFile(hFind, &FindFileData))   
{   
if (GetLastError() == ERROR\_NO\_MORE\_FILES)   
{   
return TRUE;   
}   
else   
{   
return FALSE;   
}   
}   
}

FindClose(hFind);   
}

return TRUE;   
}

void GetFileName(LPCSTR sFullPath, LPSTR sFilePath, LPSTR sFileName)   
{   
LPSTR p = (LPSTR)(sFullPath + lstrlen(sFullPath) - 1);   
bool Flag(false);

while (p != sFullPath)   
{   
if ('//' == \*p || '/' == \*p)   
{   
Flag = true;   
break;   
}   
p--;   
}

if (Flag)   
{   
lstrcpy(sFileName, p + 1);   
lstrcpy(sFilePath, sFullPath);   
sFilePath[p-sFullPath] = '/0';   
}   
else   
{   
lstrcpy(sFileName, sFullPath);   
GetFullPathName(".", MAX\_PATH, sFilePath, NULL);   
}   
}

ULONGLONG MyStartFind(LPCSTR sFindFileName)   
{   
char sPath[MAX\_PATH];   
char sFileName[MAX\_PATH];   
ULONGLONG uCountFolder(0);   
ULONGLONG uCountFile(0);   
  
GetFileName(sFindFileName, sPath, sFileName);   
MyFindFolder(sPath, sFileName, uCountFolder, uCountFile);

if (uCountFolder + uCountFile)   
{   
cout << "------------------------------------------------------------"<< endl;   
cout << "Total Folders："<< uCountFolder << "" << endl;   
cout << "Total Files： "<< uCountFile << "" << endl;   
}   
else   
cout << "Couldn't Find File." << endl;

return uCountFolder + uCountFile;   
}

int main(int argc, char \*argv[])   
{   
char sFindFileName[MAX\_PATH];   
  
if (argc < 2)   
{   
cout << "Enter Find File Name：";   
cin >> sFindFileName;   
}   
else   
{   
lstrcpy(sFindFileName, argv[1]);   
}   
MyStartFind(sFindFileName);

#ifdef \_DEBUG   
system("pause");   
#else   
if (argc < 2)   
system("pause");   
#endif

return (0);   
}

/\*Output:   
C:/a>find a   
File 1. - C:/a/a.mdb   
File 2. - C:/a/a.txt   
Folder 1. - C:/a/a1   
File 3. - C:/a/a1/a1.mdb   
File 4. - C:/a/a1/a1.txt   
Folder 2. - C:/a/a1/a2   
File 5. - C:/a/a1/a2/a2.mdb   
File 6. - C:/a/a1/a2/a2.txt   
--------------------------------   
Total Folders：2   
Total Files： 6   
\*/

**3楼**

给出方法了你就应当自己写，不过这个有点难度，所以帮你写一下，下不违例  
  
  
#include <iostream>  
#include <string>  
#include <windows.h>  
  
const std::string FGetLastError( void )  
{  
LPVOID lpMsgBuf;  
FormatMessageA(   
FORMAT\_MESSAGE\_ALLOCATE\_BUFFER |   
FORMAT\_MESSAGE\_FROM\_SYSTEM |   
FORMAT\_MESSAGE\_IGNORE\_INSERTS,  
NULL,  
GetLastError(),  
MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT),  
(LPSTR) &lpMsgBuf,  
0,  
NULL   
);  
std::string err( (LPCSTR)lpMsgBuf );  
LocalFree( lpMsgBuf );  
return err;  
}  
  
  
int main( void )  
{  
WIN32\_FIND\_DATAA ffd;  
HANDLE hff = FindFirstFileA( "C:\\\*", &ffd );  
if( hff != INVALID\_HANDLE\_VALUE )  
{  
do  
{  
if( ffd.cFileName[0] == '.' ) // .是父目录，..是本目录  
continue;  
if( ffd.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY ) // 目录  
continue; // 自己扩展吧  
  
std::cout << ffd.cFileName << " [" << ((((unsigned \_\_int64)ffd.nFileSizeHigh)<<32)|ffd.nFileSizeLow) << "]\n";  
  
} while( FindNextFileA(hff,&ffd) ); std::cout<<std::flush;  
  
if( GetLastError() != ERROR\_NO\_MORE\_FILES )  
{  
std::cerr << FGetLastError() << std::endl;  
}  
  
FindClose( hff );  
}  
else  
{  
std::cerr << FGetLastError() << std::endl;  
}  
}

关键词：

<io.h>  
\_findfirst / \_findnext / \_findclose  
  
或  
  
<windows.h>  
FindFirstFile / FindNextFile / FindClose

使用FindFirstFile（）与FindNextFile（）这两个函数即可以实现  
  
使用FindFirstFile（）函数时有个WIN32\_FIND\_DATA结构体，这个结构体里面就包含了文件的大小  
  
typedef struct \_WIN32\_FIND\_DATA {   
DWORD dwFileAttributes;   
FILETIME ftCreationTime;   
FILETIME ftLastAccessTime;   
FILETIME ftLastWriteTime;   
DWORD nFileSizeHigh;   
DWORD nFileSizeLow;   
DWORD dwOID;   
TCHAR cFileName[MAX\_PATH];   
} WIN32\_FIND\_DATA;

获得文件的属性，其中的WIN32\_FIND\_DATA结构有两个存储文件大小的字段。   
其中获得   
DWORD nFileSizeHigh;   
DWORD nFileSizeLow;  
(fileInfo.nFileSizeHigh \* (MAXDWORD+1)) + fileInfo.nFileSizeLow 就是文件大小

获取文件大小：

1、#include <stdio.h>

FILE \*fp=fopen("c:/123.bmp",rb);

fseek(fp,0,SEEK\_END);//文件指针移到尾

ulong flen=ftell(fp);//文件长度

fseek(fp,0,SEEK\_SET);//移回来

2、#include <stdio.h>

#include <io.h>

FILE \*fp=fopen("c:/123.bmp",rb);

ulong flen=filelength(fileno(fp));

3、#include <sys/stat.h>

struct \_stat info;

\_stat(filepath, &info);

int size = info.st\_size;

4、HANDLE hFile=CreateFile("c://123.bmp",GENERIC\_WRITE|GENERIC\_READ,

FILE\_SHARE\_READ,NULL,OPEN\_EXISTING,//已清空文件的方式打开

FILE\_ATTRIBUTE\_NORMAL,NULL);

DWORD dwflen=GetFileSize(hFile,NULL);

CloseHandle(hFile);

