Recursive file search using C++ MFC?

What is the cleanest way to recursively search for files using C++ and MFC?

EDIT: Do any of these solutions offer the ability to use multiple filters through one pass? I guess with CFileFind I could filter on *.* and then write custom code to further filter into different file types. Does anything offer built-in multiple filters (ie. *.exe,*.dll)?

EDIT2: Just realized an obvious assumption that I was making that makes my previous EDIT invalid. If I am trying to do a recursive search with CFileFind, I have to use *.* as my wildcard because otherwise subdirectories won't be matched and no recursion will take place. So filtering on different file-extentions will have to be handled separately regardless.

c++ | mfc | recursion | file-search

edited **May 29 '09 at 15:04**

asked **May 27 '09 at 17:12**
jacobsee
217 3 16
78% accept rate

2 The CFileFind class is just a thin wrapper over the FindFirstFile and FindNextFile Windows API functions. These don't give any provision for multiple wildcards. – Mark Ransom May 28 '09 at 16:06

feedback

5 Answers

Using `CFileFind` .

Take a look at this example from MSDN:

```
void Recurse(LPCTSTR pstr)
{
   CFileFind finder;

   // build a string with wildcards
   CString strWildcard(pstr);
   strWildcard += _T("\\*.*");

   // start working for files
   BOOL bWorking = finder.FindFile(strWildcard);

   while (bWorking)
   {
      bWorking = finder.FindNextFile();

      // skip . and .. files; otherwise, we'd
      // recur infinitely!

      if (finder.IsDots())
         continue;

      // if it's a directory, recursively search it
      if (finder.IsDirectory())
      {
         CString str = finder.GetFilePath();
         Recurse(str);
      }
}
```

feedback

Check out the recls library - stands for *rec*ursive *ls* - which is a recursive search library that works on UNIX and Windows. It's available with adaptations to different language, including C++. From memory, you can use it something like the following:

```
using recls::search_sequence;r << endl;
CString dir = "C:\\mydir";
CString patterns = "*.doc;abc*.xls";
CStringArray paths;
search_sequence files(dir, patterns, recls::RECURSIVE);

for(search_sequence::const_iterator b = files.begin(); b != files.end(); b++) {
    paths.Add((*b).c_str());
}
```

12.1k 21 57

It'll find all .doc files, and all .xls files beginning with abc in C:\mydir or any of its subdirectories.

I haven't compiled this, but it should be pretty close to the mark.

answered **May 27 '09 at 23:53**

DannyT
997 4 9

feedback

Use Boost's Filesystem implementation!

The recursive example is even on the filesystem homepage:

```
bool find_file( const path & dir_path,         // in this directory,
                const std::string & file_name, // search for this name,
                path & path_found )            // placing path here if found
{
  if ( !exists( dir_path ) ) return false;
  directory_iterator end_itr; // default construction yields past-the-end
  for ( directory_iterator itr( dir_path );
        itr != end_itr;
        ++itr )
  {
    if ( is_directory(itr->status()) )
    {
      if ( find_file( itr->path(), file_name, path_found ) ) return true;
    }
    else if ( itr->leaf() == file_name ) // see below
    {
      path_found = itr->path();
      return true;
    }
  }
  return false;
}
```

answered **May 27 '09 at 17:14**

Kieveli
4,784 15 29

feedback

I know it is not your question, but it is also easy to to without recursion by using a `CStringArray`

```
void FindFiles(CString srcFolder)
{
  CStringArray dirs;
  dirs.Add(srcFolder + "\\*.*");

  while(dirs.GetSize() > 0) {
     CString dir = dirs.GetAt(0);
     dirs.RemoveAt(0);

     CFileFind ff;
     BOOL good = ff.FindFile(dir);

     while(good) {
        good = ff.FindNextFile();
        if(!ff.IsDots()) {
          if(!ff.IsDirectory()) {
                //process file
          } else {
                //new directory (and not . or ..)
                dirs.InsertAt(0,nd + "\\*.*");
          }
        }
     }
     ff.Close();
  }
}
```

feedback

There is an old Dr Dobbs article that gives a custom CFileFindDriver class for this application. (Also talks about the classic recursion pattern using CFileFind.)

answered **May 27 '09 at 17:27**

crashmstr
7,594 2 17 33

answered **May 27 '09 at 17:14**

jacobsee
217 3 16

feedback

Not the answer you're looking for? Browse other questions tagged c++ mfc recursion file-search **or ask your own question.**

question feed