**Your current directory:**

Two functions that deal with directories are \_getcwd and the \_chdir. Both of these functions have already been written for you. All you need to do is include direct.h and io.h

Prototypes for \_getcwd and \_chdir

**int \_chdir( char \*dirpath );**

**char \*\_getcwd(char \*cstr, int size);**

\_chdir function – you will pass a c style string that contains the directory you wish to change to. If it is successful at changing to this directory, it will return a 0, any other value is an error and you should output a message to the listing.html file

**if( \_chdir( “c:\\music” ) == 0 )**

**cout << “Program changed directories successfully” << endl;**

**else**

**cout << “Unable to change to the directory: c:\\music” << endl;**

\_getcwd function – will return the pointer to a c style array that contains the name of the directory you are in. This c style array is dynamically allocated and must be freed by you. Be smart, for each directory you process, call the function only once. By passing NULL and zero forces it to due dynamic memory allocation. If you pass in a char array, the size of the array must be passed and should be of size \_MAX\_PATH. It is safer to use the dynamic method.

**char \*buffer;**

**buffer = \_getcwd( NULL, 0 );**

**cout << buffer << endl; // would output c:\music after calling the above \_chdir.**

**delete [] buffer;**

**Getting a directory listing:**

To get a directory listing, you will need to include two files that are MS Visual Studio specific.

#include <direct.h>

#include <io.h>

The following code snippet will return a complete listing of all files and folders within a given directory.

**\_finddata\_t a\_file; intptr\_t dir\_handle;**

**dir\_handle = \_findfirst( "\*.\*", &a\_file); if( dir\_handle == -1 ) return;**

**do { cout << a\_file.name << endl;**

**}while( \_findnext( dir\_handle, &a\_file ) == 0 );**

**\_findclose( dir\_handle );**

\_finddata\_t and intptr\_t are system defined data types. You do not need to create them. Just use them.

**\_finddata\_t** is a structure for a file / folder. From it you can access the fields (name, attrib, size, time\_create, time\_access, time\_write). The two fields we will be using are the name and attrib field. The name field is a C style character array of \_MAX\_PATHcharacters. \_MAX\_PATH is a system defined variable for the maximum number of characters any file path can contain. In Windows XP and Vista it is defined to be 256. The attrib is an unsigned integer that provides information about the file or folder.

A list of system attribute constants are:

**\_A\_ARCH** - Archive. Set whenever the file is changed and cleared by the **BACKUP** command. Value: 0x20.

**\_A\_HIDDEN** - Hidden file. Not normally seen with the DIR command, unless the **/AH** option is used. Returns information about normal files and files with this attribute. Value: 0x02.

**\_A\_NORMAL** - Normal. File has no other attributes set and can be read or written to without restriction. Value: 0x00.

**\_A\_RDONLY** - Read-only. File cannot be opened for writing and a file with the same name cannot be created. Value: 0x01.



**\_A\_SUBDIR** - Subdirectory. Value: 0x10.



**\_A\_SYSTEM** - System file. Not normally seen with the **DIR** command, unless the **/A** or **/A:S** option is used. Value: 0x04.

For the directory listing, you can do a bitwise and with any of the attribute constants to find out information. For instance, if you were interested in finding out if a file/folder was hidden, you would use the \_A\_HIDDEN attribute constant like so.

**if( a\_file.attrib & \_A\_HIDDEN )**

**cout << a\_file.name << “is a hidden file” << endl;**

To find out if it is a directory within the directory being processed, you would use \_A\_SUBDIR.

If the result is a number, it is a directory otherwise it is a file.