NAME

dlltool - Create files needed to build and use DLLs.

SYNOPSIS

dlltool [**-d|--input-def** *def-file-name*] [**-b|--base-file** *base-file-name*] [**-e|--output-exp** *exports-file-name*] [**-z|--output-def** *def-file-name*] [**-l|--output-lib** *library-file-name*] [**-y|--output-delaylib** *library-file-name*] [**--export-all-symbols**] [**--no-export-all-symbols**] [**--exclude-symbols** *list*] [**--no-default-excludes**] [**-S|--as** *path-to-assembler*] [**-f|--as-flags** *options*] [**-D|--dllname** *name*] [**-m|--machine** *machine*] [**-a|--add-indirect**] [**-U|--add-underscore**] [**--add-stdcall-underscore**] [**-k|--kill-at**] [**-A|--add-stdcall-alias**] [**-p|--ext-prefix-alias** *prefix*] [**-x|--no-idata4**] [**-c|--no-idata5**] [**--use-nul-prefixed-import-tables**] [**-I|--identify** *library-file-name*] [**--identify-strict**] [**-i|--interwork**] [**-n|--nodelete**] [**-t|--temp-prefix** *prefix*] [**-v|--verbose**] [**-h|--help**] [**-V|--version**] [**--no-leading-underscore**] [**--leading-underscore**] [object-file ...]

DESCRIPTION

dlltool reads its inputs, which can come from the **-d** and **-b** options as well as object files specified on the command line. It then processes these inputs and if the **-e** option has been specified it creates a exports file. If the **-l** option has been specified it creates a library file and if the **-z** option has been specified it creates a def file. Any or all of the **-e**, **-l** and **-z** options can be present in one invocation of dlltool.

When creating a DLL, along with the source for the DLL, it is necessary to have three other files. **dlltool** can help with the creation of these files.

The first file is a *.def* file which specifies which functions are exported from the DLL, which functions the DLL imports, and so on. This is a text file and can be created by hand, or **dlltool** can be used to create it using the **-z** option. In this case **dlltool** will scan the object files specified on its command line looking for those functions which have been specially marked as being exported and put entries for them in the *.def* file it creates.

In order to mark a function as being exported from a DLL, it needs to have an **-export:<name_of_function>** entry in the **.drectve** section of the object file. This can be done in C by using the *asm()* operator:

```
asm (".section .drectve");
asm (".ascii \"-export:my_func\"");

int my_func (void) { ... }
```

The second file needed for DLL creation is an exports file. This file is linked with the object files that make up the body of the DLL and it handles the interface between the DLL and the outside world. This is a binary file and it can be created by giving the **-e** option to **dlltool** when it is creating or reading in a *.def* file.

The third file needed for DLL creation is the library file that programs will link with in order to access the functions in the DLL (an 'import library'). This file can be created by giving the **-l** option to dlltool when it is creating or reading in a *.def* file.

If the **-y** option is specified, dlltool generates a delay-import library that can be used instead of the normal import library to allow a program to link to the dll only as soon as an imported function is called for the first time. The resulting executable will need to be linked to the static delayimp library containing *__delayLoadHelper2()*, which in turn will import LoadLibraryA and GetProcAddress from kernel32.

dlltool builds the library file by hand, but it builds the exports file by creating temporary files containing assembler statements and then assembling these. The **-S** command line option can be used to specify the path to the assembler that dlltool will use, and the **-f** option can be used to pass specific flags to that assembler. The **-n** can be used to prevent dlltool from deleting these temporary assembler files when it is done, and if **-n** is specified twice then this will prevent dlltool from deleting the temporary object files it used to build the library.

Here is an example of creating a DLL from a source file **dll.c** and also creating a program (from an object file called **program.o**) that uses that DLL:

```
gcc -c dll.c
dlltool -e exports.o -l dll.lib dll.o
gcc dll.o exports.o -o dll.dll
gcc program.o dll.lib -o program
```

dlltool may also be used to query an existing import library to determine the name of the DLL to which it is associated. See the description of the **-I** or **--identify** option.

OPTIONS

The command line options have the following meanings:

-d *filename*
--input-def *filename*
Specifies the name of a *.def* file to be read in and processed.

-b *filename*
--base-file *filename*
Specifies the name of a base file to be read in and processed. The contents of this file will be added to the relocation section in the exports file generated by dlltool.

-e *filename*
--output-exp *filename*
Specifies the name of the export file to be created by dlltool.

-z *filename*
--output-def *filename*
Specifies the name of the *.def* file to be created by dlltool.

-l *filename*
--output-lib *filename*
Specifies the name of the library file to be created by dlltool.

-y *filename*
--output-delaylib *filename*
Specifies the name of the delay-import library file to be created by dlltool.

--export-all-symbols
Treat all global and weak defined symbols found in the input object files as symbols to be exported. There is a small list of symbols which are not exported by default; see the **--no-default-excludes** option. You may add to the list of symbols to not export by using the **--exclude-symbols** option.

--no-export-all-symbols
Only export symbols explicitly listed in an input *.def* file or in **.drectve** sections in the input object files. This is the default behaviour. The **.drectve** sections are created by **dllexport** attributes in the source code.

--exclude-symbols *list*
Do not export the symbols in *list*. This is a list of symbol names separated by comma or colon characters. The symbol names should not contain a leading underscore. This is only meaningful when **--export-all-symbols** is used.

--no-default-excludes
When **--export-all-symbols** is used, it will by default avoid exporting certain special symbols. The current list of symbols to avoid exporting is **DllMain@12**, **DllEntryPoint@0**, **impure_ptr**. You may use the **--no-default-excludes** option to go ahead and export these special symbols. This is only meaningful when **--export-all-symbols** is used.

-S *path*
--as *path*
Specifies the path, including the filename, of the assembler to be used to create the exports file.

-f *options*
--as-flags *options*
Specifies any specific command line options to be passed to the assembler when building the exports file. This option will work even if the **-S** option is not used. This option only takes one argument, and if it occurs more than once on the command line, then later occurrences will override earlier occurrences. So if it is necessary to pass multiple options to the assembler they should be enclosed in double quotes.

-D *name*
--dll-name *name*
Specifies the name to be stored in the *.def* file as the name of the DLL when the **-e** option is used. If this option is not present, then the filename given to the **-e** option will be used as the name of the DLL.

-m *machine*
-machine *machine*
Specifies the type of machine for which the library file should be built. **dlltool** has a built in default type, depending upon how it was created, but this option can be used to override that. This is normally only useful when creating DLLs for an ARM processor, when the contents of the DLL are actually encode using Thumb instructions.

-a
--add-indirect
Specifies that when **dlltool** is creating the exports file it should add a section which allows the exported functions to be referenced without using the import library. Whatever the hell that means!

-U
--add-underscore
Specifies that when **dlltool** is creating the exports file it should prepend an underscore to the names of *all* exported symbols.

--no-leading-underscore
--leading-underscore
Specifies whether standard symbol should be forced to be prefixed, or not.

--add-stdcall-underscore
Specifies that when **dlltool** is creating the exports file it should prepend an underscore to the names of exported *stdcall* functions. Variable names and non-stdcall function names are not modified. This option is useful when creating GNU-compatible import libs for third party DLLs that were built with MS-Windows tools.

-k
--kill-at
Specifies that when **dlltool** is creating the exports file it should not append the string **@ <number>**. These numbers are called ordinal numbers and they represent another way of accessing the function in a DLL, other than by name.

-A
--add-stdcall-alias
Specifies that when **dlltool** is creating the exports file it should add aliases for stdcall symbols without **@ <number>** in addition to the symbols with **@ <number>**.

-p

--ext-prefix-alias *prefix*
Causes **dlltool** to create external aliases for all DLL imports with the specified prefix. The aliases are created for both external and import symbols with no leading underscore.

-x
--no-idata4
Specifies that when **dlltool** is creating the exports and library files it should omit the `.idata4` section. This is for compatibility with certain operating systems.

--use-nul-prefixed-import-tables
Specifies that when **dlltool** is creating the exports and library files it should prefix the `.idata4` and `.idata5` by zero an element. This emulates old gnu import library generation of `dlltool`. By default this option is turned off.

-c
--no-idata5
Specifies that when **dlltool** is creating the exports and library files it should omit the `.idata5` section. This is for compatibility with certain operating systems.

-I *filename*
--identify *filename*
Specifies that **dlltool** should inspect the import library indicated by *filename* and report, on `stdout`, the name(s) of the associated DLL(s). This can be performed in addition to any other operations indicated by the other options and arguments. **dlltool** fails if the import library does not exist or is not actually an import library. See also **--identify-strict**.

--identify-strict
Modifies the behavior of the **--identify** option, such that an error is reported if *filename* is associated with more than one DLL.

-i
--interwork
Specifies that **dlltool** should mark the objects in the library file and exports file that it produces as supporting interworking between ARM and Thumb code.

-n
--nodelete
Makes **dlltool** preserve the temporary assembler files it used to create the exports file. If this option is repeated then dlltool will also preserve the temporary object files it uses to create the library file.

-t *prefix*
--temp-prefix *prefix*
Makes **dlltool** use *prefix* when constructing the names of temporary assembler and object files. By default, the temp file prefix is generated from the pid.

-v
--verbose
Make dlltool describe what it is doing.

-h
--help
Displays a list of command line options and then exits.

-V
--version
Displays dlltool's version number and then exits.

@*file*
Read command-line options from *file*. The options read are inserted in place of the original @*file* option. If *file* does not exist, or cannot be read, then the option will be treated literally, and not removed.

Options in *file* are separated by whitespace. A whitespace character may be included in an option by surrounding the entire option in either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The *file* may itself contain additional @*file* options; any such options will be processed recursively.

SEE ALSO

The Info pages for *binutils*.

COPYRIGHT