NAME

wrc – Wine Resource Compiler

SYNOPSIS

wrc [*options*] [*inputfile...*]

DESCRIPTION

wrc compiles resources from *inputfile* into win16 and win32 compatible binary format.

The source file is preprocessed with a builtin ANSI–C compatible preprocessor before the resources are compiled. See **PREPROCESSOR** below.

wrc takes a series of *inputfile* as argument. The resources are read from standard input if no inputfile is given. If the output file is not specified with **-o**, then **wrc** will write the output to *inputfile.res* with *.rc* stripped, or to *wrc.tab.res* if no inputfile was given.

OPTIONS

–b, --target=*cpu-manufacturer*[**-***kernel*]**-***os*
: Specify the target CPU and platform on which the generated code will be built. The target specification is in the standard autoconf format as returned by **config.sub**.

–D, **––define**=*id*[=*val*]
: Define preprocessor identifier *id* to (optionally) value *val*. See also **PREPROCESSOR** below.

––debug=*nn*
: Set debug level to *nn*. The value is a bitmask consisting of 1=verbose, 2=dump internals, 4=resource parser trace, 8=preprocessor messages, 16=preprocessor scanner and 32=preprocessor parser trace.

––endianness=*e*
: Win32 only; set output byte–ordering, where *e* is one of n[ative], l[ittle] or b[ig]. Only resources in source-form can be reordered. Native ordering depends on the system on which **wrc** was built. You can see the native ordering by typing *wrc –h*.

–E
: Preprocess only. The output is written to standard output if no outputfile was selected. The output is compatible with what gcc would generate.

–h, **––help**
: Prints a summary message and exits.

–i, **––input**=*file*
: The name of the input file. If this option is not used, then **wrc** will use the first non-option argument as the input file name. If there are no non-option arguments, then **wrc** will read from standard input.

–I, **––include–dir**=*path*
: Add *path* to include search directories. *path* may contain multiple directories, separated with ':'. It is allowed to specify **–I** multiple times. Include files are searched in the order in which the **–I** options were specified.
The search is compatible with gcc, in which '<>' quoted filenames are searched exclusively via the **–I** set path, whereas the '""' quoted filenames are first tried to be opened in the current directory. Also resource statements with file references are located in the same way.

–J, **––input–format**=*format*
: Sets the input format. Valid options are 'rc' or 'rc16'. Setting the input to 'rc16' disables the recognition of win32 keywords.

–l, **––language**=*lang*
: Set default language to *lang*. Default is the neutral language 0 (i.e. "LANGUAGE 0, 0").

–m16, -m32, -m64
: Generate resources for 16-bit, 32-bit or 64-bit platforms respectively. The only difference between 32-bit and 64-bit is whether the _WIN64 preprocessor symbol is defined.

−−nostdinc
Do not search the standard include path, look for include files only in the directories explicitly specified with the **−I** option.

−−no−use−temp−file
Ignored for compatibility with *windres*.

−o, **−fo**, **−−output**=*file*
Write output to *file*. Default is **inputfile.res** with **.rc** stripped or **wrc.tab.res** if input is read from standard input.

−O, **−−output−format**=*format*
Sets the output format. The supported formats are **po**, **pot**, **res**, and **res16**. If this option is not specified, the format defaults to **res**.
In **po** mode, if an output file name is specified it must match a known language name, like **en_US.po**; only resources for the specified language are output. If no output file name is specified, a separate *.po* file is created for every language encountered in the input.

−−pedantic
Enable pedantic warnings. Notably redefinition of #define statements can be discovered with this option.

−−po-dir=*dir*
Enable the generation of resource translations based on mo files loaded from the specified directory. That directory must follow the gettext convention, in particular it must contain one *.mo* file for each language, and a LINGUAS file listing the available languages.

−r Ignored for compatibility with *rc*.

−−preprocessor=*program*
This option may be used to specify the preprocessor to use, including any leading arguments. If not specified, **wrc** uses its builtin processor. To disable preprocessing, use **--preprocessor=cat**.

−U, **−−undefine**=*id*
Undefine preprocessor identifier *id*. Please note that only macros defined up to this point are undefined by this command. However, these include the special macros defined automatically by *wrc*. See also **PREPROCESSOR** below.

−−use−temp−file
Ignored for compatibility with *windres*.

−v, **−−verbose**
Turns on verbose mode (equivalent to **-d 1**).

−−version
Print version and exit.

PREPROCESSOR

The preprocessor is ANSI−C compatible with some of the extensions of the gcc preprocessor.

The preprocessor recognizes these directives: #include, #define (both simple and macro), #undef, #if, #ifdef, #ifndef, #elif, #else, #endif, #error, #warning, #line, # (both null− and line−directive), #pragma (ignored), #ident (ignored).

The preprocessor sets by default several defines:

RC_INVOKED	set to 1
__WRC__	Major version of wrc
__WRC_MINOR__	Minor version of wrc
__WRC_PATCHLEVEL__	Patch level

Win32 compilation mode also sets _WIN32 to 1.

Special macros __FILE__, __LINE__, __TIME__ and __DATE__ are also recognized and expand to their respective equivalent.

LANGUAGE SUPPORT

Language, version and characteristics can be bound to all resource types that have inline data, such as RCDATA. This is an extension to Microsoft's resource compiler, which lacks this support completely. Only VERSIONINFO cannot have version and characteristics attached, but languages are propagated properly if you declare it correctly before the VERSIONINFO resource starts.

Example:

```
1 RCDATA DISCARDABLE
LANGUAGE 1, 0
VERSION 312
CHARACTERISTICS 876
{
        1, 2, 3, 4, 5, "and whatever more data you want"
        '00 01 02 03 04 05 06 07 08'
}
```

AUTHORS

wrc was written by Bertho A. Stultiens and is a nearly complete rewrite of the first wine resource compiler (1994) by Martin von Loewis. Additional resource−types were contributed by Ulrich Czekalla and Albert den Haan. Many cleanups by Dimitrie O. Paun in 2002-2003. Bugfixes have been contributed by many Wine developers.

BUGS

− The preprocessor recognizes variable argument macros, but does not expand them correctly.
− Error reporting should be more precise, as currently the column and line number reported are those of the next token.
− Default memory options should differ between win16 and win32.

There is no support for:
− RT_DLGINCLUDE, RT_VXD, RT_PLUGPLAY and RT_HTML (unknown format)
− PUSHBOX control is unsupported due to lack of original functionality.

Fonts are parsed and generated, but there is no support for the generation of the FONTDIR yet. The user must supply the FONTDIR resource in the source to match the FONT resources.

Bugs can be reported on the **Wine bug tracker** ⟨https://bugs.winehq.org⟩.

AVAILABILITY

wrc is part of the Wine distribution, which is available through WineHQ, the **Wine development headquarters** ⟨https://www.winehq.org/⟩.

SEE ALSO

wine(1),
Wine documentation and support ⟨https://www.winehq.org/help⟩.