Resource.exe

Jamie Doornbos, MAXIS, 2/8/00

Resource.exe is a command line utility program stored in a public bin directory (usually R:/PUB/BIN) shared by different scripts and users to operate on Sims data files. It may be compiled in the Sims workspace by choosing "Resource – Win32 debug" as the current project and then building.

The first argument to resource is the program to execute. All available programs are reported by running resource with no parameters. The second argument is the name of a resource file to operate on. (For information on resource files, see separate document Resource File Overview.) Resource uses StdResFile for most of its operations, so the file specified can be the root name of a resource file group (IFF, SPF, and DIR are included), or a single file name ending in the right extension. If the program name is entered with no file, the command line options for that program are shown with a brief description.

The legal programs are listed below. Following each program name are the accepted command line formats. Parentheses denote an argument that is replaced by an actual argument by the user. Square brackets denote optional arguments. A vertical bar denotes an exclusive choice between more than one argument.

info

resource info (file)
resource info (file) (type)
The first form lists all the resources in the given file. For each type, it prints the type and the number of resource of that type. Then for each resource, it outputs the id, the name, the size, and the language, if not the default language. The second form does the same thing but only prints the specified type.

strings

resource strings (file) [(id)] [-alias (type)] [-format]
Parses resources as a StringSet and prints out the constituent strings. If only the file is given, then all STR# resources are considered. This is the standard StringSet resource type. If id is specified, then only the resource with that id is printed. If a type is specified using –alias, then only resources of that type are considered. This is useful for viewing CTSS catalog resources, which are the same format as STR#. If –format is specified, then the index and description is printed for each string (part of the StringSet format includes a description string for each string).

getstring

resource getstring (file) (type) (id)
Parses a resource as a c-string and prints it. FWAV and GLOB types in the Sims are c-strings.

setstring

resource setstring (file) (type) (id) (string)
resource setstring (file) (type) (id) (index) (string)
The first form puts the specified string into the specified resource as a c-string (opposite of getstring). The second form loads the specified resource as a StringSet, sets the given string to the given index and saves the resource. This may be used to create new StringSet resources or edit or append existing resources.

setname

resource setname (file) (type) (id) (new name)
Sets the name of the given resource in the given file to the specified name.

remove

resource remove (file) (type)
resource remove (file) (type) (id)
The first form removes all resources from the given file of the given type. The second form removes only the specified resource. A message line is printed for each resource considered.

copy

resource copy (source file) (destination file) [(type) [(id)]] –unique|overwrite|ignore [-idoffset (offset)]

Copies resources from the source file to the destination file. If neither type nor id is specified, all resources are copied. If type is specified but not id, then all resources of that type are copied. If both are specified, then only that resource is copied. The next parameter is required and specifies one of three ways to handle duplicate resource ids. "Unique" causes an available resource id to be chosen automatically. It is not defined which id will be chosen, but the algorithm just increases the id sequentially until an unused one is found. "Overwrite" causes the resource in the destination file to be removed prior to adding the new one. "Ignore" causes the resource to not be copied at all. If an id offset is specified, then the given value is added to the id before writing the new resource.

diff

resource diff (file1) (file2)

Attempts to load each resource of each file from the other file and reports all differences found.

create

resource create (file)

Attempts to create the file. If the file already exists, an error is reported. Essentially, this just calls StdResFile::Create.

filepack

resource filepack (file) (type) (id) (input file)

Packs the given input file into the specified resource. This may be used to add BMP files as BMP_ resources.

bindump

resource bindump (file) (type) (id) (output file) [-raw]

Dumps the resource to a binary file. The resource to be dumped is given by the file, type and id. The file to dump to is given by the output file name. Part of the dump is the extra information that is not a part of the resource itself, such as the name and endian type. If "-raw" is specified, this information is not output, but only the resource itself. This may be used to dump BMP_ resources back into normal files.

binpack

resource binpack (file) (type) (id) (input file)

Creates a resource of the given type and id in the given file and with data from given input file. The input file is expected to have the format as output by bindump without the –raw option.

binprint

resource binprint (file) (type) (id)

Prints out a hex representation of the resource in the given file of the given type and id.

convertstrings

resource convertstrings (file) (id) (source file) [(new resource name)]

Converts a CST file to a STR# resource. The given source file is parsed as a CST file (strings are simply delimited by '^' characters) and converted to a StringSet and saved in a STR# resource in the given file and with the given id. If the new resource name is given, the new resource is given that name. Otherwise, the name of the CST file is used.

mergestrings

resource mergestrings (file) (id) (source file) [(new resource name)]

Same as convertstrings but only copies the source string if the target string does not yet exist. Used in converting CST files that have new strings after some strings have already been authored in the IFF file.

load

resource load (file) [(type) [(id)]]

Attempts to load the specified resources and reports any errors. Theoretically, this should never report any errors if only a file name is given. However, during a transitory debugging period on IFFResFile2, file corruption could be caught by this command.

iff_mapclear

resource iff_mapclear (iff file)

Clears the optimized map resource from the given file. For this program, the file must be an actual IFF file, and not a file group base name. The IFF map optimization is only used if a pointer to the map is found in a special location. This program simply clears that pointer so that the file is read in the pre-optimized way, from chunk to chunk. This is another command that was only used to debug problems with the map optimization.

template

resource (file) (type) (id) [(TMPL file)]

Deprecated program to use a TMPL resource to interpret given resource. TMPL resources are essentially structure layouts that may be applied to another resource to yield a readable printout of the resource field names and values. This program never did much more than the basic layouts, such as fixed size arrays. On the macintosh, resource editor utilities allowed most structure layouts to be represented and edited.