NAME

tttar – process files and ToolTalk objects in an archive

SYNOPSIS

tttar c | **t** | **x [EfhpSv]** [*tarfile*] *pathname* **. . .**

tttar c | **t** | **xfL [EhpRSv]** *tttarfile* **[[–rename** *oldname newname*] **. . .]** *pathname* **. . .**

tttar –h | **–help**

tttar –v

DESCRIPTION

The **tttar** utility has two fundamentally different modes.

- Without the **L** function modifier, **tttar** acts as a ToolTalk-aware wrapper for **tar**(1), archiving (or extracting) multiple files and their ToolTalk objects onto (or from) a single archive, called a *tarfile*.
- With the **L** function modifier, **tttar** does not invoke **tar** to archive actual files, but instead archives (or extracts) only ToolTalk objects onto (or from) a single archive, called a *tttarfile*. Since without the **L** function modifier **tttar** acts like an ToolTalk-aware **tar**(1), the description below is phrased as if the **L** function modifier is in effect. That is, the text refers to *tttarfiles* instead of *tarfiles*, and it describes archiving and de-archiving only ''the ToolTalk objects of the named files'' rather than archiving and de-archiving both ''the named files and their ToolTalk objects.''

The actions of **tttar** are controlled by the first argument, the *key*, a string of characters containing exactly one function letter from the set **ctx**, and one or more of the optional function modifiers listed under **OPERANDS**. Other arguments to **tttar** are file or directory names that specify which files to archive or extract ToolTalk objects for. By default, the appearance of a directory name refers recursively to the files and subdirectories of that directory.

A file does not have to exist for a ToolTalk object to be associated with its pathname. When **tttar** descends into a directory, it does not attempt to archive the objects associated with any files that do not exist in the directory.

When extracting from a **tar** archive that is given to **tttar** either on magnetic tape or on the standard input, the current working directory must be writable, so that the *tttarfile* can be placed there temporarily.

OPTIONS

The following options are available:

–h
–help Write a help message for invoking **tttar** and then exit.

–rename *oldname newname*
Interpret the next two arguments as an *oldname* and a *newname*, respectively, and rename any entry archived as *oldname* to *newname*. If *oldname* is a directory, then **tttar** recursively renames the entries as well. If more than one **–rename** option applies to an entry (because of one or more parent directories being renamed), the most specific **–rename** option applies.

–v Write the version number of **tttar** and then exit.

OPERANDS

The following operands are supported:

key The *key* operand consists of a function letter followed immediately by zero or more modifying letters.

The function letter is one of the following:

c Create a new archive and write the ToolTalk objects of the named files onto it.

t Write to standard output the names of all the files in the archive.

x Extract the ToolTalk objects of the named files from the archive. If a named file matches a directory with contents in the archive, this directory is (recursively) extracted. The owner and modification time of the ToolTalk objects are restored (if possible). If no *filename* arguments are given, the ToolTalk objects of all files named in the archive are extracted.

The following characters can be appended to the function letter. Appending the same character more than once produces undefined results.

f Use the next argument as the name of the *tttarfile*. If *tttarfile* is given as '–', **tttar** writes to the standard output or reads from the standard input, whichever is appropriate.

h Follow symbolic links as if they were normal files or directories. Normally, **tttar** does not follow symbolic links.

p Preserve. Restore the named files to their original modes, ignoring the present *umask* value (see **umask**(2)). The **tttar** utility also extracts setUID and sticky information for the super-user. This option is only useful with the **x** function letter, and has no meaning if the **L** function letter is given.

L Do not invoke **tar**(1). This modifier must be used with the **f** function modifier, since reading and writing an **tttar** archive directly to or from magnetic tape is unimplemented.

R Do not recurse into directories. This modifier is valid only with the **L** function modifier.

v Verbose. Write to standard error the name of each file processed, preceded by a string indicating the operation being performed, as follows:

Key Letter	**String**
c	"a "
x	"x "

The file name may be followed by additional information, such as the size of the file in the archive or file system, in an unspecified format. When used with the **t** function letter, **v** writes to standard output more information about the archive entries than just the name.

The following functions and modifiers are not supported:

- The **r** and **u** function letters of **tar**(1), for incrementally updating an archive.
- The **X** and **F** function modifiers and the –**I** option of **tar**(1), for including or excluding files from being archived based on SCCS status or being listed in a special file.
- The **w** function modifier and the –**C** option of **tar**(1), for pausing or changing directories between the files listed on the command line.
- Writing and reading *tttarfiles* (that is, archives produced with the **L** function modifier) directly to and from magnetic tape.

pathname
A pathname of a regular file or directory to be archived (when the **c** function letter is used), extracted (**x**) or listed (**t**). When *pathname* is the pathname of a directory, the action applies to all of the files and (recursively) subdirectories of that directory. When the **f** letter is used in the *key* operand, the initial *pathname* operand is interpreted as an archive name, as described previously.

tarfile
A pathname of a regular file to be read or written as an archive of files.

ttarfile
A pathname of a regular file to be read or written as an archive of ToolTalk objects.

STDIN

When the **f** modifier is used with the **t** or **x** function letter and the pathname is –, the standard input is an archive file formatted as described in **EXTENDED DESCRIPTION**. Otherwise, the standard input is not used.

INPUT FILES

The files identified by the *pathname* operands are regular files or directories. The file identified by the *tarfile* operand is a regular file formatted as described in **tar**(1). The file identified by the *tttarfile* operand is a regular file formatted as described in **EXTENDED DESCRIPTION**.

ENVIRONMENT VARIABLES

The following environment variables affect the execution of **tttar**:

LANG Provide a default value for the internationalization variables that are unset or null. If *LANG* is unset or null, the corresponding value from the implementation-specific default locale will be used. If any of the internationalization variables contains an invalid setting, the utility behaves as if none of the variables had been defined.

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_MESSAGES Determine the locale that is used to affect the format and contents of diagnostic messages written to standard error and informative messages written to standard output.

NLSPATH Determine the location of message catalogues for the processing of *LC_MESSAGES*.

TZ Determine the timezone used with date and time strings.

RESOURCES

None.

ASYNCHRONOUS EVENTS

The **tttar** utility takes the standard action for all signals.

STDOUT

When the −**h** option is used, **tttar** writes to standard output a help message in an unspecified format.

When the −**v** option is used, **tttar** writes to standard output a version number in an unspecified format.

When the **f** modifier is used with the **c** function letter and the pathname is −, the standard output is an archive file formatted as described in **EXTENDED DESCRIPTION**.

Otherwise, the standard output is not used.

STDERR

The standard error is used for diagnostic messages and the file name output described under the **v** modifier (when the **t** function letter is not used).

OUTPUT FILES

Output files are created, as specified by the archive, when the **x** function letter is used.

EXTENDED DESCRIPTION

The archive file produced and read by **tttar** is formatted as described in **tar**(1), with the addition of one extra file named **tttarfile**. (If one of the user files being archived is also named **tttarfile**, the results are unspecified.) The **tttarfile** contains all the ToolTalk *spec* information for the ToolTalk objects in the other files in the archive. The contents of **tttarfile** are written according to the referenced XDR specification (RFC 1014). The only XDR data types used are:

int A four-octet signed integer, most significant octet first

string A four-octet unsigned integer length, most significant octet first, followed by the characters of the string, followed by sufficient (0 to 3) residual zero octets to make the total number of octets a multiple of four.

The **tttarfile** starts with two integers. The first is always 1, to mark this as the header record. The second is always 1, indicating this is version 1 of the *tttarfile* format. Any future revisions of the *tttarfile* format should increment the version number so older programs processing the *tttarfile* can diagnose the incompatiblity.

The end of the **tttarfile** is a integer 3, marking the end-of-file record.

In between, there is one logical record for each spec. Each logical record starts with an integer 2, marking it as a spec record. Other integer values are reserved for assignment to future data types.

After the record identifier, the spec record contains, in sequence:

1. A string giving the Tooltalk object identifier (*objid*) of the object represented by the spec
2. A string giving the name of the file (as found in the archive table of contents) that contains the contents of the ToolTalk object represented by the spec
3. A string giving the ToolTalk object type identifier (*otid*) of the ToolTalk object represented by the spec
4. An integer giving the number of properties for this object

The properties of the object immediately follow the number of properties. Each property consists of:

1. A string giving the name of the property
2. An integer, which is always zero (for historical compatibility)
3. An integer giving the number of values for this property
4. A string for each value

After the values, the next property is found, until all properties for the object have been accounted for; then the next spec is found, until all specs for objects associated with files in the archive are accounted for.

EXIT STATUS

The following exit values are returned:

0 All files and ToolTalk objects were moved successfully.

>0 An error occurred or the invoked **tar**(1) command exited with a non-zero value.

CONSEQUENCES OF ERRORS

Default.

FILES

/*mountpoint*/TT_DB The directory used as a database for the ToolTalk objects of files in the file system mounted at /*mountpoint*.

APPLICATION USAGE

None.

EXAMPLES

None.

SEE ALSO

tar(1), **ttcp**(1), **ttsession**(1).