

# pkgcheck Utility, v2.0.1

Manfred Lotz

February 7, 2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>pkgcheck utility</b>	<b>3</b>
<b>3</b>	<b>Requirements</b>	<b>4</b>
<b>4</b>	<b>Installation</b>	<b>4</b>
<b>5</b>	<b>Utility usage</b>	<b>4</b>
5.1	Help option . . . . .	4
5.2	Check a package . . . . .	5
5.3	Check a package which has a TDS archive . . . . .	5
5.4	Pkgcheck messages . . . . .	5
5.5	Duplicate files . . . . .	6
5.6	Permissions . . . . .	6
5.7	CRLF line endings . . . . .	6
5.8	Help options . . . . .	6
<b>6</b>	<b>About checking file types</b>	<b>7</b>
<b>7</b>	<b>About permissions checking</b>	<b>7</b>
<b>8</b>	<b>Different kind of messages</b>	<b>8</b>
<b>9</b>	<b>Informational messages</b>	<b>9</b>
9.1	I0001 -- Successfully converted from CRLF to LF . . . . .	9
9.2	I0002 -- Checking package files in directory . . . . .	9
9.3	I0003 -- Checking TDS zip archive . . . . .	9
9.4	I0004 -- Correcting line endings for file . . . . .	9
9.5	I0005 -- Corrections permissions for file or directory . . . . .	9

## Contents

9.6	I0006 -- Files having one of the following file name endings are regarded as temporary . . . . .	9
9.7	I0007 -- Successfully corrected wrong line endings to LF resp. CRLF . . . . .	9
<b>10</b>	<b>Warning messages</b>	<b>10</b>
10.1	W0001 -- Archive as package file detected . . . . .	10
10.2	W0002 -- Duplicate files detected . . . . .	10
10.3	W0003 -- Same named files detected in the package tree . . . . .	10
10.4	W0004 -- encoding with BOM detected . . . . .	10
10.5	W0005 -- Very large file with size <size> detected in package . . . . .	10
10.6	W0006 -- Very large file with size <size> detected in TDS zip archive . . . . .	10
10.7	W0007 -- Empty directory detected in the TDS zip archive . . . . .	11
10.8	W0008 -- Windows file has Unix line endings . . . . .	11
<b>11</b>	<b>Error messages</b>	<b>11</b>
11.1	E0001 -- Bad characters in file name . . . . .	11
11.2	E0002 -- File Permissions . . . . .	11
11.3	E0003 -- README is not a text file . . . . .	11
11.4	E0004 -- Empty directory not allowed . . . . .	11
11.5	E0005 -- Empty files not allowed . . . . .	12
11.6	E0006 -- Hidden directories not allowed . . . . .	12
11.7	E0007 -- Hidden files not allowed . . . . .	12
11.8	E0008 -- Temporary file detected . . . . .	12
11.9	E0009 -- Package doesn't contain a README file . . . . .	12
11.10	E0010 -- Broken symlink detected . . . . .	12
11.11	E0011 -- Wrong permission for directory . . . . .	12
11.12	E0012 -- CRLF line endings detected . . . . .	13
11.13	E0013 -- Socket special file detected . . . . .	13
11.14	E0014 -- Fifo special file detected . . . . .	13
11.15	E0015 -- Bloch device file detected . . . . .	13
11.16	E0016 -- Character device file detected . . . . .	13
11.17	E0017 -- PDF document is in error . . . . .	13
11.18	E0018 -- Unwanted directory detected . . . . .	13
11.19	E0019 -- Generated file detected . . . . .	14
11.20	E0021 -- Error when reading a file . . . . .	14
11.21	E0022 -- Check of an URL in a README file failed . . . . .	14
11.22	E0023 -- Follow up error when trying to read a directory with insufficient permissions . . . . .	14
11.23	E0024 -- TDS zip archive has wrong permissions . . . . .	14
11.24	E0025 -- Duplicate names when ignoring letter case for files or directories . . . . .	14
11.25	E0026 -- Files not in TDS or different in TDS and non-install tree . . . . .	15
11.26	E0027 -- An I/O error occurred . . . . .	15
11.27	E0028 -- A path name in a TDS zip archive must contain the package name . . . . .	15

## 1 Introduction

11.28	E0029 -- README file: encoding with BOM detected . . . . .	15
11.29	E0030 -- A symlink was found which points outside of the package directory tree . . . . .	15
11.30	E0031 -- File name contains invalid UTF-8 character(s) . . . . .	15
11.31	E0033 -- Error when unpacking tds archive . . . . .	15
11.32	E0034 -- Unwanted file detected in the top level directory in TDS zip archive . . . . .	16
11.33	E0035 -- Unwanted TDS archive detected in package directory tree	16
11.34	E0036 -- .dtx/.ins files found in wrong directory in TDS zip archive	16
11.35	E0037 -- CR line endings detected . . . . .	16
11.36	E0038 -- File has inconsistent line endings: CR: x, LF: y, CRLF: z .	16
<b>12</b>	<b>Fatal messages</b>	<b>16</b>
12.1	F0001 -- Specify a directory to check (use option -d) . . . . .	16
12.2	F0002 -- Specified directory does not exist. Exiting...	16
12.3	F0003 -- Specified TDS archive does not exist or is no file . . . . .	17
12.4	F0004 -- The file specified as TDS archive is no zip archive . . . . .	17
12.5	F0005 -- Bad file name for the zip archive . . . . .	17
12.6	F0006 -- Unknown error code specified with option -e resp. --- explain. Exiting... . . . . .	17
12.7	F0007 -- Could not create temporary directory for unzipping the TDS zip archive . . . . .	17

## Abstract

This document describes the pkgcheck command line utility which is used by the author when checking uploaded packages to CTAN.

## 1 Introduction

Uploaded packages to CTAN must satisfy various requirements in order to get installed on CTAN.

A first introduction is given here <https://ctan.org/help/upload-pkg>.

Even more details are to be found in the excellent CTAN-upload addendum <https://ctan.org/file/help/ctan/CTAN-upload-addendum> written by Petra Rübe-Pugliese.

The pkgcheck utility which runs on Linux systems only checks those requirements which can be checked by a program.

## 2 pkgcheck utility

The pkgcheck utility is a compiled program written in the Rust programming language. It runs in a Linux environment. Currently, Windows is not supported. Simply, because the author doesn't use Windows at all.

### 3 Requirements

It will be invoked from the command line, and any error or warning message has a certain message id. pkgcheck offers an option to get more information for a certain error.

## 3 Requirements

pkgcheck doesn't have any special runtime requirements.

The pkgcheck is a 64-bit statically linked binary, and should work on any 64-bit Linux. It is available in the repository in directory bin/.

Currently, the only external programs required are:

- /usr/bin/unzip  
Used only, when a TDS zip archive will be extracted.
- /usr/bin/pdftinfo  
Used only, when a PDF document will be checked.

## 4 Installation

Copy the binary from bin/pkgcheck to a suitable location on your hard disk, and (recommended) make sure the directory is in the PATH or call pkgcheck using an absolute path name.

## 5 Utility usage

### 5.1 Help option

Running pkgcheck --help shows the available command line options.

Here a sample output:

```
pkgcheck 1.0.0
Manfred Lotz <manfred@ctan.org>
A checker for uploaded packages to CTAN.
```

USAGE:

```
pkgcheck [FLAGS] [OPTIONS]
```

FLAGS:

-L, --correct-crlf	Correct CRLF line endings
-C, --correct-perms	Correct permissions
--explain-all	Explains all error or warning messages
-h, --help	Prints help information

## 5 Utility usage

-I, --ignore-dupes	Ignore dupes
--no-colors	Don't display messages in color
--show-temp-endings	Show file endings for temporary files
--urlcheck	Check URLs found in README files
-V, --version	Prints version information
-v, --verbose	Verbose operation?

### OPTIONS:

-e, --explain <explain>	Explain error or warning message
-d, --package-dir <pkg_dir>	Package directory
-T, --tds-zip <tds_zip>	tds zip archive

## 5.2 Check a package

A package for CTAN is supposed to be uploaded as a ZIP or a g-zipped tar archive. The package must have a top level directory.

After unpacking the archive of a package mypkg into directory mypkg/ it can be checked by running pkgcheck with option verb|-package-dir| or shorter -d.

```
1 pkgcheck -d mypkg
```

pkgcheck returns 1 if there are any errors, otherwise 0.

## 5.3 Check a package which has a TDS archive

If a package contains a TDS ZIP archive it is supposed to be in the top level directory of a package.

In order to check the TDS ZIP archive the option -T <tds\_zip> or --tds-zip <tds\_zip> can be used.

Please note that a TDS ZIP archive will always be checked together with the non-install tree of the package which means that --tds-zip requires option --package-dir as well.

Checking package mypkg pkgcheck will be invoked like follows:

```
1 pkgcheck -d mypkg -T mypkg.tds.zip
```

As before pkgcheck returns 1 if there are any errors, otherwise 0.

## 5.4 Pkgcheck messages

pkgcheck issues three kind of messages

- Information messages

## 5 Utility usage

- Warning messages
- Error messages

Messages have unique ids and the detailed explanation of a message can be either looked up in this document, or it can be displayed by using command line option `--explain` or `-e`.

1 pkgcheck -explain e0012 Example

### 5.5 Duplicate files

By default, pkgcheck detects duplicate files in a package. This could be disabled by using command line switch `--ignore-dups` or shorter `-I`.

### 5.6 Permissions

pkgcheck offers the option `--correct-crlf` or shorter `-L` to correct wrong permissions in a package.

### 5.7 CRLF line endings

pkgcheck detects CRLF line endings in text files as good as it can. It reads up to 1 MB to check for CRLF line endings.

Option `--correct-crlf` or for short `-L` can be used to convert a file from CRLF to LF line endings.

### 5.8 Help options

- `-V`  
Outputs pkgcheck's version number.
- `--help`  
`--help` shows the available command line options.

```
pkgcheck 1.0.0
Manfred Lotz <manfred@ctan.org>
A checker for uploaded packages to CTAN.
```

```
USAGE:
  pkgcheck [FLAGS] [OPTIONS]
```

```
FLAGS:
  -L, --correct-crlf      Correct CRLF line endings
```

## 6 About checking file types

-C, --correct-perms	Correct permissions
--explain-all	Explains all error or warning messages
-h, --help	Prints help information
-I, --ignore-dupes	Ignore dupes
--no-colors	Don't display messages in color
--show-temp-endings	Show file endings for temporary files
--urlcheck	Check URLs found in README files
-V, --version	Prints version information
-v, --verbose	Verbose operation?

### OPTIONS:

-e, --explain <explain>	Explain error or warning message
-d, --package-dir <pkg_dir>	Package directory
-T, --tds-zip <tds_zip>	tds zip archive

- --explain <explain>

This option explains an error message in more detail.

Example:

```
1 pkgcheck -e e0012
```

- --explain-all

Outputs a list of explanations of all messages.

- --show-temp-endings

Outputs a list of all file name endings which pkgcheck uses to detect temporary files.

## 6 About checking file types

pkgcheck determines, similar to the UNIX `file` command, the type of file. This is required before for example checking permissions or complaining that a text file has CRLF line endings.

It is very important to note that determining file types is not bullet proof. So, it might happen in some cases that pkgcheck makes mistakes when determining a file type. This could lead to a subsequent mistake when complaining about an x-bit, or complaining about CRF line ending.

## 7 About permissions checking

From an installation point of view the files and directories of a package

## 8 Different kind of messages

- must be at least world readable
- must be writable by owner or group
- must not have the x-bit on for the owner if the file isn't an executable, i.e. a script or binary

The reason for this minimal requirement is that the installation utility used by the CTAN team (which by the way was written by Rainer Schöpf a long time ago) sets permissions correctly if the owner permission is set correctly.

Examples:

- `README.md` with 666 is ok because the installation utility converts the permission to 664
- `README.md` with 660 is wrong because the installation utility wouldn't have access to the file
- some `.pdf` with 744 would be wrong because a PDF document must not have the x-bit on for the owner

Because of the smartness of the installation utility `pkgcheck` does check minimal requirements only, i.e some weird looking permissions like the 666 above are accepted.

## 8 Different kind of messages

### **Innnn** Informational messages

These are message which announce `pkgcheck` actions.

### **Fnnnn** Fatal messages

Fatal messages report unrecoverable errors. In this case `pkgcheck`'s only option is to terminate. If for example, the package directory specified at the command line doesn't exist then the only option is to terminate.

### **Ennnn** Error messages

Error messages report errors which must be fixed before installing a package.

### **Wnnnn** Warning messages

Warning messages denote possible errors depending upon the situation. For example, for a font package having many duplicate files might be ok. For another package it could be regarded as an error.



## 9 Informational messages

### 9.1 I0001 -- Successfully converted from CRLF to LF

Just an information that pkgcheck has successfully converted a file from CRLF to LF line endings

### 9.2 I0002 -- Checking package files in directory

Just an information that pkgcheck starts checking the package files in the unzipped directory trees

### 9.3 I0003 -- Checking TDS zip archive

Just an information that pkgcheck starts checking the TDS zip archive

### 9.4 I0004 -- Correcting line endings for file

The file had CRLF line ending and will be corrected to have LF (Unix like) line endings.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#crlf>

### 9.5 I0005 -- Corrections permissions for file or directory

pkgcheck corrects wrong permissions for package files and directories. It runs the chmod command in verbose mode.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#filepermissions>

### 9.6 I0006 -- Files having one of the following file name endings are regarded as temporary

Option --show-temp-endings was used, and pkgcheck prints a list of temporary file endings and their meanings.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noaux-files>

### 9.7 I0007 -- Successfully corrected wrong line endings to LF resp. CRLF

pkgcheck successfully converted wrong line endings to LF line endings or to CRLF line endings if it the file was a Windows text file.

Wrong line endings could be CR, CRLF or a mixture of line endings.

## 10 Warning messages

### 10.1 W0001 -- Archive as package file detected

Usually a CTAN package should not contain archives. An exception are situations where, for example, the source code of a package is kept in a separate zip archive.

### 10.2 W0002 -- Duplicate files detected

Duplicate files were detected which are listed right after this message.

The message is a warning message as something like this could not be seen as an error in general.

### 10.3 W0003 -- Same named files detected in the package tree

We like to have unique file names over the whole package directory tree. When we discover same named files we report it as a warning. Common names like README, README.txt, README.md, Makefile, Makefile.in, Makefile.am and makefile are ignored when checking.

For more details refer to: <http://mirror.utexas.edu/ctan/help/ctan/CTAN-upload-addendum.html#uniquefilenames>

### 10.4 W0004 -- encoding with BOM detected

A UTF encoded package file contains a BOM (byte order mark). Currently, we issues a warning.

Nevertheless, the CTAN team discourages uses of BOM. Please be aware, that in some future time this could be reagarded as an error.

### 10.5 W0005 -- Very large file with size <size> detected in package

(Experimental) We issue the message if there is a file is larger than 40MiB in the package directory tree.

### 10.6 W0006 -- Very large file with size <size> detected in TDS zip archive

(Experimental) We issue the message if there is a file larger than 40MiB in the TDS zip archive.

### 10.7 W0007 -- Empty directory detected in the TDS zip archive

Empty directories in a TDS zip archive are discouraged. As they usually don't create errors in the distribution we issue a warning only.

### 10.8 W0008 -- Windows file has Unix line endings

A Windows file with Unix line endings was detected.

We regard a file as a Windows file if its name ends with:

- .bat
- .cmd
- .nsh, or
- .reg

## 11 Error messages

### 11.1 E0001 -- Bad characters in file name

File name should not contain non-ascii characters. Additionally, file names should not contain control characters or other characters which may have a special meaning for UNIX shells.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#nunixspecialcharacters>

### 11.2 E0002 -- File Permissions

Files submitted to CTAN should be world readable.

Only files that are truly executable (like scripts and binaries) should be marked as such.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#filepermissions>

### 11.3 E0003 -- README is not a text file

The README file specified in the error message must be a text file but it isn't.

### 11.4 E0004 -- Empty directory not allowed

Empty directories are considered as rubbish, and are usually not accepted as part of a package, neither in the package tree nor in the TDS zip archive.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noemptyfiles>

### 11.5 E0005 -- Empty files not allowed

Empty files are considered as rubbish, and are usually not accepted as part of a package.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noemptyfiles>

### 11.6 E0006 -- Hidden directories not allowed

A package should not contain hidden directories, neither in the package tree nor in the TDS zip archive.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noaux-files>

### 11.7 E0007 -- Hidden files not allowed

A package should not contain hidden files, neither in the package tree nor in the TDS zip archive.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noaux-files>

### 11.8 E0008 -- Temporary file detected

A temporary file was detected. These are typically files created by TeX & friends and should not be part of a package.

Temporary files will also be detected in a TDS zip archive.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#noaux-files>

### 11.9 E0009 -- Package doesn't contain a README file

A package must contain at least one of README, README.md or README.txt file.

For more details refer to: <http://mirrors.ibiblio.org/CTAN/help/ctan/CTAN-upload-addendum.html#readme>

### 11.10 E0010 -- Broken symlink detected

A broken symlink was detected.

### 11.11 E0011 -- Wrong permission for directory

Directories should have rwx for the owner and at least r-x for others (i.e. world readable).

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#filepermissions>

### **11.12 E0012 -- CRLF line endings detected**

The file specified in the error message contains CRLF line endings. Text files should have UNIX style line endings.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#crlf>

### **11.13 E0013 -- Socket special file detected**

The file specified in the error message is a socket special file which is not allowed.

### **11.14 E0014 -- Fifo special file detected**

The file specified in the error message is a fifo special file which is not allowed.

### **11.15 E0015 -- Block device file detected**

The file specified in the error message is a block device file which is not allowed.

### **11.16 E0016 -- Character device file detected**

The file specified in the error message is a character device file which is not allowed.

### **11.17 E0017 -- PDF document is in error**

The PDF document mentioned in the message is in error.

pdfinfo will be run to check if a PDF document can be read. Message E0017 will be followed by the error messages from pdfinfo.

Example:

```
I0002   Checking package files in directory somepkg
E0017   PDF error detected in somepkg/sompkg.pdf
Syntax Error (1293042): Illegal character ')'
Syntax Error: Couldn't find trailer dictionary
Syntax Error (1293042): Illegal character ')'
Syntax Error: Couldn't find trailer dictionary
Syntax Error: Couldn't read xref table
```

### **11.18 E0018 -- Unwanted directory detected**

A directory was detected which should not be part of a package. Example: \_\_MACOSX

### **11.19 E0019 -- Generated file detected**

In order to avoid redundancy we don't want to have included files in a package which easily can be generated from other files in the submission.

Exceptions are the README files of the package, i.e. README, README.md or README.txt.

Starting with version 1.1.0 pkgcheck also detects generated files if they are in a different directory in the package.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#no-generatedfiles>

### **11.20 E0021 -- Error when reading a file**

An error was encountered when reading the file specified in the message.

### **11.21 E0022 -- Check of an URL in a README file failed**

URL checking is in effect. An error occurred when trying to retrieve an URL which was found in the specified README file.

### **11.22 E0023 -- Follow up error when trying to read a directory with insufficient permissions**

Error which is a follow-up error. For instance, when a directory could not be read.

### **11.23 E0024 -- TDS zip archive has wrong permissions**

The TDS zip archive should have at least `r--` for the owner and at least `r--` for others (i.e. world readable).

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#filepermissions>

### **11.24 E0025 -- Duplicate names when ignoring letter case for files or directories**

As there are operating systems which do not distinguish between `myfile` and `MYFILE` we don't want to have file names in a directory which are the same after converting to lower case.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#filenames>

### **11.25 E0026 -- Files not in TDS or different in TDS and non-install tree**

The file mentioned in the error message is either not existing in the TDS zip archive, or it is different to the one in the non-install tree

### **11.26 E0027 -- An I/O error occurred**

Some kind of I/O error occurred. If you believe there is an error in pkgcheck please contact the author.

### **11.27 E0028 -- A path name in a TDS zip archive must contain the package name**

The path names in a TDS zip archive must contain the package name.

**Example:** Assume a package somepkg. Then path names should look like follows:

```
tex/latex/somepkg/somepkg.cls
doc/latex/somepkg/README
source/latex/somepkg/somepkg.dtx
...
```

### **11.28 E0029 -- README file: encoding with BOM detected**

A README file should be either ASCII or UTF-8 without BOM(byte order mark)

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#readme>

### **11.29 E0030 -- A symlink was found which points outside of the package directory tree**

A symlink must not point to a file or directory outside of the package directory tree.

### **11.30 E0031 -- File name contains invalid UTF-8 character(s)**

A file name contains invalid UTF-8 character(s).

### **11.31 E0033 -- Error when unpacking tds archive**

In order to investigate the contents of the TDS zip archive pkgcheck unpacks the TDS zip archive to a temporary location which failed for the reason given in the error message.

## 12 Fatal messages

### 11.32 E0034 -- Unwanted file detected in the top level directory in TDS zip archive

A top level directory of a TDS archive should only contain certain directories but no files.

### 11.33 E0035 -- Unwanted TDS archive detected in package directory tree

A package directory should not contain a TDS zip archive.

### 11.34 E0036 -- .dtx/.ins files found in wrong directory in TDS zip archive

In a TDS zip archive a .dtx resp. .ins file must be in a subdirectory of either of source/ or doc/ top level directories.

### 11.35 E0037 -- CR line endings detected

The file specified in the error message contains CR line endings. Text files should have UNIX style line endings.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#crlf>

### 11.36 E0038 -- File has inconsistent line endings: CR: x, LF: y, CRLF: z

The file specified in the error message contains CR line endings. Text files should have UNIX style line endings.

For more details refer to: <http://mirror.ctan.org/help/ctan/CTAN-upload-addendum.html#crlf>

## 12 Fatal messages

### 12.1 F0001 -- Specify a directory to check (use option -d)

pkgcheck was called without any options. Use option -d to check a directory

### 12.2 F0002 -- Specified directory does not exist. Exiting...

The directory specified at the command line does not exist.



**12.3 F0003 -- Specified TDS archive does not exist or is no file**

Specify a valid TDS zip archive when calling pkgcheck

**12.4 F0004 -- The file specified as TDS archive is no zip archive**

Specify a valid TDS zip archive when calling pkgcheck

**12.5 F0005 -- Bad file name for the zip archive**

pkgcheck detected that the file name of the TDS zip archive doesn't end with .tds.zip

**12.6 F0006 -- Unknown error code specified with option -e resp. ---explain. Exiting...**

pkgcheck was called with option -e resp. --explain, and an unknown error code was specified.

**12.7 F0007 -- Could not create temporary directory for unzipping the TDS zip archive**

Make sure the temp directory is writable and/or the temp directory has enough space.