NAME

utfcheck − Check a file to verify that it is valid UTF-8 or ASCII

SYNOPSIS

utfcheck [−a] [−q] [−−expurgated] [−i *input_file.beta*] [−o *output_file.utf8*]

DESCRIPTION

utfcheck(1) reads an input file and prints messages about contents that might be unexpected (even if legal Unicode) in a UTF-8 or ASCII file, such as embedded control characters or Unicode "noncharacters". No diagnostic messages are printed for the control characters horizontal tab, vertical tab, line feed, or form feed. A final summary will indicate if null, carriage return, or escape characters were read.

utfcheck will detect a UTF-16 big-endian or little-endian Byte Order Mark at the beginning of a file and quit if it sees one. There is no support for parsing UTF-16 files beyond initial detection of the Byte Order Mark.

OPTIONS

−a Test for a pure ASCII file. ASCII control characters are allowed, but **utfcheck** will fail if it encounters a byte with value greater than hexadecimal 7F (the delete control character).

−i Specify the input file. The default is STDIN.

−o Specify the output file. The default is STDOUT.

−q Quiet mode. Do not print any output unless an illegal byte sequence is detected.

−−expurgated

Check a UTF-8 file against the "expurgated" version of the Unicode Standard, the one without the Byte Order Mark, after Monty Python's "Bookshop" skit with the "expurgated" version of *Olsen's Standard Book of British Birds*, the one without the gannet—because the customer didn't like them. (But they've all got the Byte Order Mark. It's a standard part of the Unicode Standard, the Byte Order Mark. It's in all the books.) This option is not abbreviated, to keep the user mindful of the questionable nature of testing for the lack of something even though it is a legitimate part of the Unicode Standard. **utfcheck** will fail if this option is selected and the UTF-8 Byte Order Mark (officially the zero width no-break space) is detected anywhere in the input file.

Sample usage:

utfcheck −i *my_input_file.txt* −o *my_output_file.log*

MESSAGES

IMMEDIATE MESSAGES

Some uncommon characters are noted immediately as they are encountered. Some are fatal errors and some are not, as noted below. The messages associated with them follow.

ASCII-CONTROL: U+*nnnn*

The file contains ASCII control characters in the range U+0001 through U+001F, inclusive, except for Horizontal Tab, Line Feed, Vertical Tab, Form Feed, New Line, Carriage Return; or the file contains the Delete character (U+007F).

ASCII-NULL

The file contains an ASCII NULL character (U+0000).

BINARY-DATA: 0x*nn*

The file contains a byte value that is not part of a well-formed UTF-8 character. This is considered a fatal error and the program will terminate with exit status EXIT_FAILURE.

NON-ASCII-DATA: 0x*nn*

The **−a** (ASCII only) option was selected and the file contains non-ASCII data (i.e., a byte with the high bit set). This is considered a fatal error and the program will terminate with exit status EXIT_FAILURE.

SURROGATE-PAIR-CODE-POINT: 0x*nn*... (U+*nnnn*)
The file contains a Unicode surrogate pair code point encoded as UTF-8 (U+D800 through U+DFFF, inclusive). Surrogate code points are used with UTF-16 files, so they should never appear in UTF-8 files. The byte values are printed first, and then the UTF-8 converted Unicode code point is printed in parentheses. This is considered a fatal error and the program will terminate with exit status EXIT_FAILURE.

UTF-16-BE: Unsupported
The file begins with a big-endian UTF-16 Byte Order Mark. Because **utfcheck** does not support UTF-16, this is considered a fatal error and the program will terminate with exit status EXIT_FAILURE.

UTF-16-LE: Unsupported
The file begins with a little-endian UTF-16 Byte Order Mark. Because **utfcheck** does not support UTF-16, this is considered a fatal error and the program will terminate with exit status EXIT_FAILURE.

UTF-8-BOM-BEGIN
The file begins with a Byte Order Mark (U+FEFF) in UTF-8 form. If the **−−expurgated** option is selected and this condition is detected, this is considered a fatal error and the program will terminate with exit status EXIT_FAILURE; otherwise, the program continues.

UTF-8-BOM-EMBEDDED
The file contains a Byte Order Mark (U+FEFF) after the start of the file. If the **−−expurgated** option is selected and this condition is detected, this is considered a fatal error and the program will terminate with exit status EXIT_FAILURE; otherwise, the program continues.

UTF-8-CONTROL: 0x*nn*... (U+*nnnn*)
The file contains a UTF-8 control character (U+0080 through U+009F, inclusive). The byte values are printed first, and then the UTF-8 converted Unicode code point is printed in parentheses.

UTF-8-NONCHARACTER: 0x*nn*... (U+*nnnn*)
The file contains a Unicode "noncharacter". This can be a code point in the range U+FDD0 through U+FDEF, inclusive, or the last two code points of any Unicode plane, from Plane 0 through Plane 16, inclusive. The byte values are printed first, and then the UTF-8 converted Unicode code point is printed in parentheses. Note that a noncharacter is allowable in well-formed Unicode files, so this condition is not considered an error.

END OF FILE SUMMARY

If the **−q** option is not selected and the program has not encountered a fatal error before reaching the end of the input stream, **utfcheck** prints a summary of the file contents after the input stream has reached its end. This will begin with the line "FILE-SUMMARY:". This is followed by a line beginning with "Character-Set: " followed by one of "ASCII", "UTF-8", "UTF-16-BE" (UTF-16 Big Endian), "UTF-16-LE" (UTF-16 Little Endian), or "BINARY". (Note that UTF-16 parsing is not currently implemented, so the UTF-16-BE and UTF-16-LE types will not appear in this final summary at present.) The following messages can appear in this end of file summary if the program encountered the corresponding types of Unicode code points.

BOM-AT-START
The file begins with a UTF-8 Byte Order Mark (U+FEFF).

BOM-AFTER-START
The file contains a UTF-8 Byte Order Mark (U+FEFF) after the start of the file.

CONTAINS-NULLS
The file contains null characters (U+0000).

CONTAINS-CARRIAGE_RETURN
The file contains carriage returns (U+000D).

CONTAINS-CONTROL_CHARACTERS
The file contains ASCII control characters in the range U+0001 through U+001F, inclusive, except for Horizontal Tab, Line Feed, Vertical Tab, Form Feed, New Line, or Carriage Return; or contains the Delete character (U+007F) or control characters in the range U+0080 through U+009F, inclusive.

CONTAINS-ESCAPE_SEQUENCES
The file contains at least one ASCII escape character (U+001B), which is interpreted to be part of an escape sequence (for example, a VT-100 or ANSI terminal control sequence).

Plane-0-PUA: *n* **characters**
Number of Plane 0 Private Use Area characters in file.

Plane-15-PUA: *n* **characters**
Number of Plane 15 Private Use Area characters in file.

Plane-16-PUA: *n* **characters**
Number of Plane 16 Private Use Area characters in file.

EXIT STATUS

utfcheck will exit with a status of EXIT_SUCCESS if the input file only contains valid text, or with a status of EXIT_FAILURE if it contains invalid bytes.

FILES

ASCII or UTF-8 text files.

AUTHOR

utfcheck was written by Paul Hardy.

LICENSE

BUGS

No known bugs exist.