NAME

Dpkg::Control::HashCore - parse and manipulate a block of RFC822-like fields

DESCRIPTION

The Dpkg::Control::Hash object is a hash-like representation of a set of RFC822-like fields. The fields names are case insensitive and are always capitalized the same when output (see field_capitalize function in Dpkg::Control::Fields). The order in which fields have been set is remembered and is used to be able to dump back the same content. The output order can also be overridden if needed.

You can store arbitrary values in the hash, they will always be properly escaped in the output to conform to the syntax of control files. This is relevant mainly for multilines values: while the first line is always output unchanged directly after the field name, supplementary lines are modified. Empty lines and lines containing only dots are prefixed with " ." (space + dot) while other lines are prefixed with a single space.

During parsing, trailing spaces are stripped on all lines while leading spaces are stripped only on the first line of each field.

FUNCTIONS

my `$c` = Dpkg::Control::Hash->new(%opts)

Creates a new object with the indicated options. Supported options are:

allow_pgp
: Configures the parser to accept OpenPGP signatures around the control information. Value can be 0 (default) or 1.

allow_duplicate
: Configures the parser to allow duplicate fields in the control information. Value can be 0 (default) or 1.

drop_empty
: Defines if empty fields are dropped during the output. Value can be 0 (default) or 1.

name
: The user friendly name of the information stored in the object. It might be used in some error messages or warnings. A default name might be set depending on the type.

is_pgp_signed
: Set by the parser (starting in dpkg 1.17.0) if it finds an OpenPGP signature around the control information. Value can be 0 (default) or 1, and undef when the option is not supported by the code (in versions older than dpkg 1.17.0).

`$c`->set_options($option, `%opts`)

Changes the value of one or more options.

my `$value` = `$c`->get_option($option)

Returns the value of the corresponding option.

`$c`->load($file)

Parse the content of `$file`. Exits in case of errors. Returns true if some fields have been parsed.

`$c`->parse_error($file, `$fmt`, ...)

Prints an error message and dies on syntax parse errors.

`$c`->parse($fh, `$description`)

Parse a control file from the given filehandle. Exits in case of errors. `$description` is used to describe the filehandle, ideally it's a filename or a description of where the data comes from. It's used in error messages. Returns true if some fields have been parsed.

`$c->`find_custom_field($name)
: Scan the fields and look for a user specific field whose name matches the following regex: /X[SBC]*-$name/i. Return the name of the field found or undef if nothing has been found.

`$c->`get_custom_field($name)
: Identify a user field and retrieve its value.

`$c->`save($filename)
: Write the string representation of the control information to a file.

my `$str` = `$c->`*output()*
"$c"
: Get a string representation of the control information. The fields are sorted in the order in which they have been read or set except if the order has been overridden with *set_output_order()*.

`$c->`output($fh)
: Print the string representation of the control information to a filehandle.

`$c->`set_output_order(@fields)
: Define the order in which fields will be displayed in the *output()* method.

`$c->`apply_substvars($substvars)
: Update all fields by replacing the variables references with the corresponding value stored in the Dpkg::Substvars object.

CHANGES

Version 1.01

New method: `$c->`*parse_error()*.

Version 1.00

Mark the module as public.

AUTHOR

Raphal Hertzog <hertzog@debian.org>.