



Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'Mail::Field.3pm'

\$ man Mail::Field.3pm

Mail::Field(3pm) User Contributed Perl Documentation Mail::Field(3pm)

NAME

Mail::Field - base-class for manipulation of mail header fields

INHERITANCE

Mail::Field is extended by

Mail::Field::AddrList

Mail::Field::Date

Mail::Field::Generic

SYNOPSIS

```
use Mail::Field;

my $field = Mail::Field->new('Subject', 'some subject text');
my $field = Mail::Field->new(Subject => 'some subject text');
print $field->tag,"- ",$field->stringify,"\n";

my $field = Mail::Field->subject('some subject text');
```

DESCRIPTION

"Mail::Field" creates and manipulates fields in MIME headers, collected within a Mail::Header object. Different field types have their own sub-class (extension), defining additional useful accessors to the field content.

People are invited to merge their implementation to special fields into MailTools, to maintain a consistent set of packages and documentation.

METHODS

Constructors

Mail::Field (and it's sub-classes) define several methods which return new objects. These

can all be categorized as constructor.

`Mail::Field->combine($fields)`

Take a LIST of "Mail::Field" objects (which should all be of the same sub-class) and create a new object in that same class.

`Mail::Field->extract($tag, $head [, $index])`

Takes as arguments the tag name, a "Mail::Head" object and optionally an index.

If the index argument is given then "extract" will retrieve the given tag from the "Mail::Head" object and create a new "Mail::Field" based object. undef will be returned in the field does not exist.

If the index argument is not given the result depends on the context in which "extract" is called. If called in a scalar context the result will be as if "extract" was called with an index value of zero. If called in an array context then all tags will be retrieved and a list of "Mail::Field" objects will be returned.

`Mail::Field->new($tag [, STRING | %options])`

Create an object in the class which defines the field specified by the \$tag argument.

"Fake" constructors

`$obj->create(%options)`

This constructor is used internally with preprocessed field information. When called on an existing object, its original content will get replaced.

`$obj->parse()`

Parse a field line.

Accessors

`$obj->set(%options)`

Change the settings (the content, but then smart) of this field.

`$obj->stringify()`

Returns the field as a string.

`$obj->tag()`

`Mail::Field->tag()`

Return the tag (in the correct case) for this item. Well, actually any casing is OK, because the field tags are treated case-insensitive; however people have some preferences.

Smart accessors

`$obj->text([STRING])`

Without arguments, the field is returned as `stringify()` does. Otherwise, the `STRING` is parsed with `parse()` to replace the object's content.

It is more clear to call either `stringify()` or `parse()` directly, because this method does not add additional processing.

DETAILS

SUB-CLASS PACKAGE NAMES

All sub-classes should be called `Mail::Field::name` where `name` is derived from the tag using these rules.

- ? Consider a tag as being made up of elements separated by '-'
- ? Convert all characters to lowercase except the first in each element, which should be uppercase.
- ? `name` is then created from these elements by using the first `N` characters from each element.
- ? `N` is calculated by using the formula :-

$$\text{int}((7 + \#\text{elements}) / \#\text{elements})$$

- ? `name` is then limited to a maximum of 8 characters, keeping the first 8 characters.

For an example of this take a look at the definition of the `"_header_pkg_name()`" subroutine in `"Mail::Field"`

DIAGNOSTICS

Error: Undefined subroutine `<method>` called

`Mail::Field` objects use autoloading to compile new functionality. Apparently, the method called is not implemented for the specific class of the field object.

SEE ALSO

This module is part of the MailTools distribution, <http://perl.overmeer.net/mailtools/>.

AUTHORS

The MailTools bundle was developed by Graham Barr. Later, Mark Overmeer took over maintenance without commitment to further development.

`Mail::Cap` by Gisle Aas aas@oslonett.no. `Mail::Field::AddrList` by Peter Orbaek poe@cit.dk. `Mail::Mailer` and `Mail::Send` by Tim Bunce Tim.Bunce@ig.co.uk. For other contributors see `ChangeLog`.

LICENSE

Copyrights 1995-2000 Graham Barr gbarr@pobox.com and 2001-2017 Mark Overmeer perl@overmeer.net.

This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself. See <http://www.perl.com/perl/misc/Artistic.html>

perl v5.28.1

2019-07-25

Mail::Field(3pm)