NAME

Mail::Message::Head::SpamGroup − spam fighting related header fields

INHERITANCE

```
Mail::Message::Head::SpamGroup
  is a Mail::Message::Head::FieldGroup
  is a Mail::Reporter
```

SYNOPSIS

```
my $sg = Mail::Message::Head::SpamGroup->new(head => $head, ...);
$head->addSpamGroup($sg);

my $sg = $head->addSpamGroup( <options> );
$sg->delete;

my @sgs = $head->spamGroups;
```

DESCRIPTION

A *spam group* is a set of header fields which are added by spam detection and spam fighting software. This class knows various details about that software.

Extends ‘‘DESCRIPTION’’ in Mail::Message::Head::FieldGroup.

METHODS

Extends ‘‘METHODS’’ in Mail::Message::Head::FieldGroup.

Constructors

Extends ‘‘Constructors’’ in Mail::Message::Head::FieldGroup.

`$obj`->**clone**()

Inherited, see ‘‘Constructors’’ in Mail::Message::Head::FieldGroup

`$obj`->**fighter**(`$name`, [$settings])

Mail::Message::Head::SpamGroup->**fighter**(`$name`, [$settings])

Get the `$settings` of a certain spam-fighter, optionally after setting them. The **knownFighters()** method returns the defined names. The names are case-sensitive.

```
-Option --Default
 fields   <required>
 isspam   <required>
 version  undef
```

fields => REGEXP

The regular expression which indicates which of the header fields are added by the spam fighter software.

isspam => CODE

The CODE must return true or false, to indicate whether the spam fighter thinks that the message contains spam. The CODE ref is called with the spamgroup object (under construction) and the header which is inspected.

version => CODE

Can be called to collect the official name and the version of the software which is used to detect spam. The CODE ref is called with the spamgroup object (under construction) and the header which is inspected.

example: adding your own spam-fighter definitions

```
Mail::Message::Head::SpamGroup->fighter( 'MY-OWN',
   fields => qw/^x-MY-SPAM-DETECTOR-/,
   isspam => sub { my ($sg, $head) = @_; $head->fields > 100 }
  );
```

`$obj->`**from**($head|$message, `%options`)
Returns a list of `Mail::Message::Head::SpamGroup` objects, based on the specified `$message` or message `$head`.

```
-Option--Default
 types   undef
```

types => ARRAY-OF-NAMES
Only the specified types will be tried. If the ARRAY is empty, an empty list is returned. Without this option, all sets are returned.

`$obj->`**habeasSweFieldsCorrect**([$message|$head])
Mail::Message::Head::SpamGroup->**habeasSweFieldsCorrect**([$message|$head])
Returns a true value if the `$message` or `$head` contains `Habeas-SWE` fields which are correct. Without argument, this is used as instance method on an existing Spam-Group.

example: checking Habeas-SWE fields

```
if(Mail::Message::Head::SpamGroup->habeasSweFieldsCorrect($message))
{   $message->label(spam => 0);
}

my $sg = $message->head->spamGroups('Habeas-SWE');
if($sg->habeasSweFieldsCorrect) { ... };

use List::Util 'first';
if(first {$_->habeasSweFieldsCorrect} $head->spamGroups)
{   ...
}
```

`$obj->`**implementedTypes**()
Mail::Message::Head::SpamGroup->**implementedTypes**()
Inherited, see ‘‘Constructors’’ in Mail::Message::Head::FieldGroup

`$obj->`**isSpamGroupFieldName**($name)
Mail::Message::Head::SpamGroup->**isSpamGroupFieldName**($name)
`$obj->`**knownFighters**()
Mail::Message::Head::SpamGroup->**knownFighters**()
Returns an unsorted list of all names representing pre-defined spam-fighter software. You can ask details about them, and register more fighters with the **fighter()** method.

Mail::Message::Head::SpamGroup->**new**($fields, `%options`)
Construct an object which maintains one set of fields which were added by spam fighting software.

```
-Option  --Defined in                      --Default
 head      Mail::Message::Head::FieldGroup  undef
 log       Mail::Reporter                   'WARNINGS'
 software  Mail::Message::Head::FieldGroup  undef
 trace     Mail::Reporter                   'WARNINGS'
 type      Mail::Message::Head::FieldGroup  undef
 version   Mail::Message::Head::FieldGroup  undef
```

head => HEAD
log => LEVEL
software => STRING
trace => LEVEL
type => STRING
version => STRING

`$obj->`**spamDetected**([BOOLEAN])
Returns (after setting) whether this group of spam headers thinks that this is spam. See **Mail::Message::Head::Complete::spamDetected()**.

example:

```
die if $head->spamDetected;

foreach my $sg ($head->spamGroups)
{   print $sg->type." found spam\n" if $sg->spamDetected;
}
```

The header
Extends ‘‘The header’’ in Mail::Message::Head::FieldGroup.

`$obj->`**add**(<$field, `$value`> | `$object`)
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**addFields**([$fieldnames])
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**attach**($head)
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**delete**()
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**fieldNames**()
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**fields**()
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

`$obj->`**head**()
Inherited, see ‘‘The header’’ in Mail::Message::Head::FieldGroup

Access to the header
Extends ‘‘Access to the header’’ in Mail::Message::Head::FieldGroup.

`$obj->`**software**()
Inherited, see ‘‘Access to the header’’ in Mail::Message::Head::FieldGroup

`$obj->`**type**()
Inherited, see ‘‘Access to the header’’ in Mail::Message::Head::FieldGroup

`$obj->`**version**()
Inherited, see ‘‘Access to the header’’ in Mail::Message::Head::FieldGroup

Internals
Extends ‘‘Internals’’ in Mail::Message::Head::FieldGroup.

`$obj->`**collectFields**([$name])
Inherited, see ‘‘Internals’’ in Mail::Message::Head::FieldGroup

`$obj->`**detected**($type, `$software`, `$version`)
Inherited, see ‘‘Internals’’ in Mail::Message::Head::FieldGroup

Error handling
Extends ‘‘Error handling’’ in Mail::Message::Head::FieldGroup.

`$obj->`**AUTOLOAD**()
Inherited, see ‘‘Error handling’’ in Mail::Reporter

`$obj->`**addReport**($object)
Inherited, see ‘‘Error handling’’ in Mail::Reporter

`$obj->`**defaultTrace**([$level]|[$loglevel, `$tracelevel`]|[$level, `$callback`])
Mail::Message::Head::SpamGroup->**defaultTrace**([$level]|[$loglevel, `$tracelevel`]|[$level, `$callback`])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**details**()
Inherited, see ''Error handling'' in Mail::Message::Head::FieldGroup

`$obj->`**errors**()
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**log**([$level, [$strings]])
Mail::Message::Head::SpamGroup->**log**([$level, [$strings]])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**logPriority**($level)
Mail::Message::Head::SpamGroup->**logPriority**($level)
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**logSettings**()
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**notImplemented**()
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**print**([$fh])
Inherited, see ''Error handling'' in Mail::Message::Head::FieldGroup

`$obj->`**report**([$level])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**reportAll**([$level])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**trace**([$level])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**warnings**()
Inherited, see ''Error handling'' in Mail::Reporter

Cleanup

Extends ''Cleanup'' in Mail::Message::Head::FieldGroup.

`$obj->`**DESTROY**()
Inherited, see ''Cleanup'' in Mail::Reporter

DETAILS

Spam fighting fields

Detected spam fighting software

The Mail::Message::Head::SpamGroup class can be used to detect fields which were produced by different spam fighting software.

- SpamAssassin

  These fields are added by Mail::SpamAssassin, which is the central implementation of the spamassassin package. The homepage of this GPL'ed project can be found at <http://spamassassin.org>.

- Habeas-SWE

  Habeas tries to fight spam via the standard copyright protection mechanism: Sender Warranted E-mail (SWE). Only when you have a contract with Habeas, you are permitted to add a few copyrighted lines to your e-mail. Spam senders will be refused a contract. Mail clients which see these nine lines are (quite) sure that the message is sincere.

  See <http://www.habeas.com> for all the details on this commercial product.

- MailScanner

  The MailScanner filter is developed and maintained by transtec Computers. The software is available for free download from <http://www.sng.ecs.soton.ac.uk/mailscanner/>. Commercial support is provided via <http://www.mailscanner.biz>.

DIAGNOSTICS

Error: Package `$package` does not implement `$method`.

Fatal error: the specific package (or one of its superclasses) does not implement this method where it should. This message means that some other related classes do implement this method however the class at hand does not. Probably you should investigate this and probably inform the author of the package.

SEE ALSO

This module is part of Mail-Message distribution version 3.012, built on February 11, 2022. Website: *http://perl.overmeer.net/CPAN/*

LICENSE