NAME

Mail::Box::Thread::Manager − maintain threads within a set of folders

INHERITANCE

```
Mail::Box::Thread::Manager
  is a Mail::Reporter
```

SYNOPSIS

```
my $mgr     = Mail::Box::Manager−>new;
my $folder  = $mgr−>open(folder => '/tmp/inbox');

my $threads = $mgr−>threads();
$threads−>includeFolder($folder);

my $threads = $msg−>threads(folder => $folder);

foreach my $thread ($threads−>all) {
    $thread−>print;
}

$threads−>removeFolder($folder);
```

DESCRIPTION

A (message−)*thread* is a message with links to messages which followed in reply of that message. And then the messages with replied to the messages, which replied the original message. And so on. Some threads are only one message long (never replied to), some threads are very long.

The `Mail::Box::Thread::Manager` is very powerful. Not only is it able to do a descent job on MH-like folders (makes a trade-off between perfection and speed), it also can maintain threads from messages residing in different opened folders. Both facilities are rare for mail-agents. The manager creates flexible trees with Mail::Box::Thread::Node objects.

Extends "DESCRIPTION" in Mail::Reporter.

METHODS

Extends "METHODS" in Mail::Reporter.

Constructors

Extends "Constructors" in Mail::Reporter.

Mail::Box::Thread::Manager−>**new**(%options)

A `Mail::Box::Thread::Manager` object is usually created by a Mail::Box::Manager. One manager can produce more than one of these objects. One thread manager can combine messages from a set of folders, which may be partially overlapping with other objects of the same type.

```
 −Option     −−Defined in     −−Default
  dummy_type                    Mail::Message::Dummy
  folder                        [ ]
  folders                       [ ]
  log          Mail::Reporter   'WARNINGS'
  thread_body                   <false>
  thread_type                   Mail::Box::Thread::Node
  timespan                      '3 days'
  trace        Mail::Reporter   'WARNINGS'
  window                        10
```

dummy_type => CLASS

The type of dummy messages. Dummy messages are used to fill holes in detected threads: referred to by messages found in the folder, but itself not in the folder.

folder => FOLDER | REF-ARRAY-FOLDERS
: Specifies which folders are to be covered by the threads. You can specify one or more open folders. When you close a folder, the manager will automatically remove the messages of that folder from your threads.

folders => FOLDER | REF-ARRAY-FOLDERS
: Equivalent to the `folder` option.

log => LEVEL
thread_body => BOOLEAN
: May thread-detection be based on the content of a message? This has a serious performance implication when there are many messages without `In−Reply−To` and `References` headers in the folder, because it will cause many messages to be parsed. NOT IMPLEMENTED YET.

thread_type => CLASS
: Type of the thread nodes.

timespan => TIME | ’EVER’
: Specify how fast threads usually work: the amount of time between an answer and a reply. This is used in combination with the `window` option to determine when to give-up filling the holes in threads.

: See **Mail::Box::timespan2seconds()** for the possibilities for TIME. With ’EVER’, the search for messages in a thread will only be limited by the window-size.

trace => LEVEL
window => INTEGER|’ALL’
: The thread-window describes how many messages should be checked at maximum to fill ‘holes’ in threads for folder which use delay-loading of message headers.

: The constant ’ALL’ will cause thread-detection not to stop trying to fill holes, but continue looking until the first message of the folder is reached. Gives the best quality results, but may perform bad.

example:

```
use Mail::Box::Manager;
my $mgr     = Mail::Box::Manager−>new;
my $inbox   = $mgr−>open(folder => $ENV{MAIL});
my $read    = $mgr−>open(folder => 'Mail/read');
my $threads = $mgr−>threads(folders => [$inbox, $read]);

# longer alternative for last line:
my $threads = $mgr−>threads;
$threads−>includeFolder($inbox);
$threads−>includeFolder($read);
```

Grouping Folders

`$obj−>`**folders**()
: Returns the folders as managed by this threader.

`$obj−>`**includeFolder**($folders)
: Add one or more folders to the list of folders whose messages are organized in the threads maintained by this object. Duplicated inclusions will not cause any problems.

: From the folders, the messages which have their header lines parsed (see Mail::Box about lazy extracting) will be immediately scanned. Messages of which the header is known only later will have to report this (see **toBeThreaded()**).

example:

```
$threads−>includeFolder($inbox, $draft);
```

`$obj−>`**removeFolder**($folders)

Remove one or more folders from the list of folders whose messages are organized in the threads maintained by this object.

example:

```
 $threads−>removeFolder($draft);
```

The Threads

`$obj−>`**all**()

Returns all messages which start a thread. The list may contain dummy messages and messages which are scheduled for deletion.

To be able to return all threads, thread construction on each message is performed first, which may be slow for some folder-types because is will enforce parsing of message-bodies.

`$obj−>`**known**()

Returns the list of all messages which are known to be the start of a thread. Threads containing messages which where not read from their folder (like often happens MH-folder messages) are not yet known, and hence will not be returned.

The list may contain dummy messages, and messages which are scheduled for deletion. Threads are detected based on explicitly calling **inThread()** and **thread()** with a messages from the folder.

Be warned that, each time a message's header is read from the folder, the return of the method can change.

`$obj−>`**sortedAll**([$prepare, [$compare]])

Returns **all()** the threads by default, but sorted on timestamp.

`$obj−>`**sortedKnown**([$prepare, [$compare]])

Returns all **known()** threads, in sorted order. By default, the threads will be sorted on timestamp, But a different `$compare` method can be specified.

`$obj−>`**thread**($message)

Returns the thread where this `$message` is the start of. However, there is a possibility that this message is a reply itself.

Usually, all messages which are in reply of this message are dated later than the specified one. All headers of messages later than this one are getting parsed first, for each folder in this threads-object.

example:

```
 my $threads = $mgr−>threads(folder => $inbox);
 my $thread  = $threads−>thread($inbox−>message(3));
 print $thread−>string;
```

`$obj−>`**threadStart**($message)

Based on a message, and facts from previously detected threads, try to build solid knowledge about the thread where this message is in.

Internals

`$obj−>`**createDummy**($message_id)

Get a replacement message to be used in threads. Be warned that a dummy is not a member of any folder, so the program working with threads must test with **Mail::Message::isDummy()** before trying things only available to real messages.

`$obj−>`**inThread**($message)

Collect the thread-information of one message. The 'In−Reply−To' and 'Reference' header-fields are processed. If this method is called on a message whose header was not read yet (as usual for MH-folders, for instance) the reading of that header will be triggered here.

`$obj->`**outThread**($message)
Remove the message from the thread-infrastructure. A message is replaced by a dummy.

`$obj->`**toBeThreaded**($folder, `@messages`)
Include the specified messages in/from the threads managed by this object, if this folder is maintained by this thread-manager.

`$obj->`**toBeUnthreaded**($folder, `@messages`)
Remove the specified `@messages` in/from the threads managed by this object, if this folder is maintained by this thread-manager.

Error handling

Extends "Error handling" in Mail::Reporter.

`$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::Box::Thread::Manager−>**defaultTrace**([$level]|[$loglevel, `$tracelevel`]|[$level, `$callback`])
Inherited, see "Error handling" in Mail::Reporter

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

`$obj->`**log**([$level, [$strings]])
Mail::Box::Thread::Manager−>**log**([$level, [$strings]])
Inherited, see "Error handling" in Mail::Reporter

`$obj->`**logPriority**($level)
Mail::Box::Thread::Manager−>**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

`$obj->`**report**([$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::Reporter.

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

DETAILS

This module implements thread-detection on a folder. Messages created by the better mailers will include `In−Reply−To` and `References` lines, which are used to figure out how messages are related. If you prefer a better thread detection, they are implementable, but there may be a serious performance hit (depends on the type of folder used).

Maintaining threads

A `Mail::Box::Thread::Manager` object is created by the Mail::Box::Manager, using **Mail::Box::Manager::threads()**. Each object can monitor the thread-relations between messages in one or more folders. When more than one folder is specified, the messages are merged while reading the threads, although nothing changes in the folder-structure. Adding and removing folders which have to be maintained is permitted at any moment, although may be quite costly in performance.

An example of the maintained structure is shown below. The Mail::Box::Manager has two open folders, and a thread-builder which monitors them both. The combined folders have two threads, the second is two long (msg3 is a reply on msg2). Msg2 is in two folders at once.

```
manager
 |    \
 |     `----------- threads
 |                  |     |
 |                thread thread---thread
 |                  |    /|        /
 |                  |   //        /
 +---- folder1      |  //        /
 |       |         /  //        /
 |       `-----msg1  //        /
 |       `-----msg2-'/        /
 |                  /        /
 `-----folder2     /        /
         |        /        /
         `-----msg2       /
         `-----msg3------'
```

Delayed thread detection

With **all()** you get the start-messages of each thread of this folder. When that message was not found in the folder (not saved or already removed), you get a message of the dummy-type. These thread descriptions are in perfect state: all messages of the folder are included somewhere, and each missing message of the threads (*holes*) are filled by dummies.

However, to be able to detect all threads it is required to have the headers of all messages, which is very slow for some types of folders, especially MH and IMAP folders.

For interactive mail-readers, it is preferred to detect threads only on messages which are in the viewport of the user. This may be sloppy in some situations, but everything is preferable over reading an MH mailbox with 10k e−mails to read only the see most recent messages.

In this object, we take special care not to cause unnecessary parsing (loading) of messages. Threads will only be detected on command, and by default only the message headers are used.

The following reports the Mail::Box::Thread::Node which is related to a message:

```
 my $thread = $message->thread;
```

When the message was not put in a thread yet, it is done now. But, more work is done to return the best thread. Based on various parameters, which where specified when the folder was created, the method walks through the folder to fill the holes which are in this thread.

Walking from back to front (recently arrived messages are usually in the back of the folder), message after message are triggered to be included in their thread. At a certain moment, the whole thread of the requested method is found, a certain maximum number of messages was tried, but that didn't help (search window bound reached), or the messages within the folder are getting too old. Then the search to complete the thread will end, although more messages of them might have been in the folder: we don't scan the whole folder for performance reasons.

Finally, for each message where the head is known, for instance for all messages in mbox-folders, the correct thread is determined immediately. Also, all messages where the head get loaded later, are automatically included.

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-Box distribution version 3.009, built on August 18, 2020. Website: *http://perl.overmeer.net/CPAN/*

LICENSE