NAME

Mail::Box::Tie::ARRAY − access an existing message folder as array

SYNOPSIS

```
use Mail::Box::Manager;
my $mgr    = Mail::Box::Manager->new;
my $folder = $mgr->open(folder => 'inbox');

use Mail::Box::Tie::ARRAY;
tie my(@inbox), 'Mail::Box::Tie::ARRAY', $folder;

# deprecated, but works too
use Mail::Box::Tie;
tie my(@inbox), 'Mail::Box::Tie', $folder;

foreach (@inbox) {print $_->short}
print $_->print foreach @inbox;
my $emails = @inbox;

print $inbox[3];
print scalar @inbox;
push @inbox, Mail::Box::Message->new(...);
delete $inbox[6];
print $inbox[0]->head->get('status');

my $folder = tied @inbox;
untie @inbox;
```

DESCRIPTION

Certainly when you look at a folder as a list of messages, it is logical to access the folder through an array.

Not all operations on arrays are supported. Actually, most functions which would reduce the size of the array are modified instead to mark messages for deletion.

Examples what you *cannot* do:

```
shift/unshift/pop/splice @inbox;
```

METHODS

Constructors

TIEARRAY('Mail::Box::Tie::ARRAY', FOLDER)

Create the tie on an existing folder.

example: tie an array to a folder

```
my $mgr   = Mail::Box::Manager->new;
my $inbox = $mgr->new(folder => $ENV{MAIL});
tie my(@inbox), 'Mail::Box::Tie::Array', ref $inbox, $inbox;
```

Tied Interface

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

Flag a message to be removed. Be warned that the message stays in the folder, and is not removed before the folder is written.

example:

```
delete $inbox[5];
$inbox[5]->delete;   #same
```

`$obj−>`**FETCH**($index)

Get the message which is at the indicated location in the list of messages contained in this folder. Deleted messages will be returned as `undef`.

example:

```
 print $inbox[3];     # 4th message in the folder
 print @inbox[3,0];   # 4th and first of the folder
 print $inbox[−1];    # last message
```

`$obj−>`**FETCHSIZE**()

Return the total number of messages in a folder. This is called when the folder-array is used in scalar context, for instance.

example:

```
 if(@inbox > 10)    # contains more than 10 messages?
 my $nrmsgs = @inbox;
```

`$obj−>`**PUSH**(@messages)

Add `@messages` to the end of the folder.

example:

```
    push @inbox, $newmsg;
```

`$obj−>`**STORE**($index, `$message`)

Random message replacement is not permitted −−doing so would disturb threads etc. An error occurs if you try to do this. The only thing which is allowed is to store a message at the first free index at the end of the folder (which is also achievable with **PUSH()**).

example:

```
 $inbox[8] = $add;
 $inbox[−1] = $add;
 push @inbox, $add;
```

`$obj−>`**STORESIZE**($length)

Sets all messages behind from `$length` to the end of folder to be deleted.

DETAILS

Folder tied as array

Limitations

This module implements `TIEARRAY`, `FETCH`, `STORE`, `FETCHSIZE`, `STORESIZE`, `DELETE`, `PUSH`, and `DESTROY`.

This module does not implement all other methods as described in the Tie::Array documentation, because the real array of messages is not permitted to shrink or be mutilated.

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