NAME

Mail::Box::Tie::HASH − access an existing message folder as a hash

SYNOPSIS

```
tie my(%inbox), 'Mail::Box::Tie::HASH', $folder;

foreach my $msgid (keys %inbox)
{   print $inbox{$msgid};
    delete $inbox{$msgid};
}

$inbox{$msg->messageId} = $msg;
```

DESCRIPTION

Certainly when you look at a folder as being a set of related messages based on message-id, it is logical to access the folder through a hash.

For a tied hash, the message-id is used as the key. The message-id is usually unique, but when two or more instances of the same message are in the same folder, one will be flagged for deletion and the other will be returned.

This implementation uses basic folder access routines which are related to the message-id.

METHODS

Constructors

TIEHASH('Mail::Box::Tie::HASH', FOLDER)

Connects the FOLDER object to a HASH.

example:

```
my $mgr    = Mail::Box::Manager->new;
my $folder = $mgr->open(access => 'rw');
tie my(%inbox), 'Mail::Box::Tie::HASH', $folder;
```

Tied Interface

$obj->**CLEAR**()

Remove the contents of the hash. This is not really possible, but all the messages will be flagged for deletion.

example:

```
%inbox = ();
%inbox = ($msg->messageId, $msg); #before adding msg
```

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

Remove the message with the specified `$message_id`.

example:

```
delete $inbox{$msgid};
```

$obj->**EXISTS**($message_id)

Check whether a message with a certain `$message_id` exists.

example:

```
if(exists $inbox{$msgid}) ...
```

$obj->**FETCH**($message_id)

Get the message with the specified id. The returned message may be a dummy if message thread detection is used. Returns `undef` when there is no message with the specified id.

example:

```
my $msg = $inbox{$msgid};
if($inbox{$msgid}->isDummy)  ...
```

$obj->**FIRSTKEY**()
See **NEXTKEY()**.

$obj->**NEXTKEY**($previous)
FIRSTKEY() returns the first message-id/message pair from the folder, and NEXTKEY returns the message-id/message pair for the next message, in the order in which the message is stored in the folder.

Messages flagged for deletion will **not** be returned. See the **Mail::Box::messages()** method of the folder type for more information about the folder message order.

example:

```
foreach my $msgid (keys %inbox) ...
foreach my $msg (values %inbox) ...

while(my ($msgid, $msg) = each %inbox) {
   $msg->print unless $msg->isDeleted;
}
```

$obj->**STORE**(undef, `$message`)
Store a message in the folder. The key must be `undef`, because the message-id of the specified message is taken. This is shown in the first example. However, as you see, it is a bit complicated to specify `undef`, therefore the string `"undef"` is accepted as well.

The message may be converted into something which can be stored in the folder type which is at stake. The added instance is returned.

example:

```
$inbox{ (undef) } = $msg;
$inbox{undef} = $msg;
```

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