NAME

Mail::Box::Identity − represents an unopened folder

INHERITANCE

```
Mail::Box::Identity
  is an User::Identity::Item

Mail::Box::Identity
  is a Mail::Reporter
```

SYNOPSIS

```
use User::Identity;
use Mail::Box::Identity;
my $me   = User::Identity->new(...);

my $mailbox = Mail::Box::Identity->new(...);
$me->add(folders => $mailbox);

# Simpler

use User::Identity;
my $me   = User::Identity->new(...);
my $addr = $me->add(folders => ...);
```

DESCRIPTION

The `Mail::Box::Identity` object contains the description of a single mailbox. The mailboxes are collected by an Mail::Box::Collection object. This corresponds with IMAP's `\NoSelect`, for instance.

Nearly all methods can return undef.

Extends ''DESCRIPTION'' in Mail::Reporter.

Extends ''DESCRIPTION'' in User::Identity::Item.

METHODS

Extends ''METHODS'' in Mail::Reporter.

Extends ''METHODS'' in User::Identity::Item.

Constructors

Extends ''Constructors'' in Mail::Reporter.

Extends ''Constructors'' in User::Identity::Item.

Mail::Box::Identity->**new**([$name], `%options`)

```
 -Option     --Defined in          --Default
  deleted                          <false>
  description  User::Identity::Item  undef
  folder_type                      from parent
  inferiors                        1
  location                         undef
  log          Mail::Reporter      'WARNINGS'
  manager                          <from parent>
  marked                           undef
  name         User::Identity::Item  <required>
  only_subs                        <foldertype and name dependent>
  parent       User::Identity::Item  undef
  subf_type                        <same as parent>
  trace        Mail::Reporter      'WARNINGS'
```

deleted => BOOLEAN
The folder is flagged for deletion. This not have any implications yet, because it may still get undeleted.

description => STRING
folder_type => CLASS
inferiors => BOOLEAN
Can this folder have children? If not, this is cleared.

location => DIRECTORY|FILENAME
The location of this folder. Often, only the manager can figure-out where this folder really is.

log => LEVEL
manager => OBJECT
Any Mail::Box::Manager or Mail::Box::Manage::User OBJECT.

marked => BOOLEAN|`undef`
Whether the folder is flagged for some reason, for instance because new messages have arrived.

name => STRING
only_subs => BOOLEAN
Some folder types can have messages in their toplevel folder, other cannot. That determines the default. See**Mail::Box::topF olderWithMessages()**

parent => OBJECT
subf_type => CLASS
The type for a subfolder collection, must extend CLASS Mail::Box::Collection.

trace => LEVEL

Attributes

Extends ‘‘Attributes’’ in User::Identity::Item.

`$obj`->**deleted**([BOOLEAN])
`$obj`->**description**()
Inherited, see ‘‘Attributes’’ in User::Identity::Item

`$obj`->**folderType**()
Returns the type of this folder.

`$obj`->**fullname**([$delimeter])
Returns the name of the folder, from the toplevel until this one, with the `$delimeter` string between each level. `$delimeter` default to a forward slash (a `/`).

`$obj`->**inferiors**([BOOLEAN])
`Inferiors` are `subfolders`. When this flag is set, it is permitted to create subfolders.

`$obj`->**location**([$filename|$directory|undef])
Returns the directory or filename of the folder. If this is not pre-defined, it is computed based on the knowledge about the folder type. Be sure to set the location of the toplevel folder to the folderdir of the user to get this to work.

`$obj`->**manager**()
Returns the manager (usually a Mail::Box::Manage::User which owns the folders. May be undefined, by default from parent.

`$obj`->**marked**([BOOLEAN|undef])
When something special has happened with the folder, this flag can be set (or cleared). The `undef` status is an ‘‘unknown’’. In the IMAP4 protocol, `0` will result in a `\Unmarked`, a `1` results in a `\Marked`, and `undef` in nothing.

`$obj`->**name**([$newname])
Inherited, see ‘‘Attributes’’ in User::Identity::Item

`$obj->`**onlySubfolders**([BOOLEAN])
Than this folder be opened (without trying) or not? The default depends on the folder type, and whether this is the toplevel folder or not. See **Mail::Box::topFolderWithMessages()**

`$obj->`**topfolder**()
Run up the tree to find the highest level folder.

Collections
Extends ‘‘Collections’’ in User::Identity::Item.

`$obj->`**add**($collection, `$role`)
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**addCollection**(`$object` | <[$type], `%options`>)
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**collection**($name)
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**parent**([$parent])
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**removeCollection**($object|$name)
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**type**()
Mail::Box::Identity->**type**()
Inherited, see ‘‘Collections’’ in User::Identity::Item

`$obj->`**user**()
Inherited, see ‘‘Collections’’ in User::Identity::Item

Searching
Extends ‘‘Searching’’ in User::Identity::Item.

`$obj->`**find**($collection, `$role`)
Inherited, see ‘‘Searching’’ in User::Identity::Item

Subfolders
`$obj->`**addSubfolder**($m<Mail::Box::Identity>|$data)
Add a new folder into the administration. With `$data`, a new object will be instantiated first. The identity is returned on success.

`$obj->`**folder**([..., `$name`])
Returns the subfolder’s object with `$name` or `undef` if it does not exist. When multiple NAMEs are added, those super folders are traverst first. Without any `$name`, the current object is returned

example: get some folder

```
my $a = $user->folders->folder('b', 'a');

my $name  = "a:b:c";
my $delim = ":";
my $f = $user->folders->folder(split $delim, $name);
```

`$obj->`**foreach**(CODE)
For each of the subfolders found below this point call CODE. This current folder is called first. Be warned that you may find identities with the **deleted()** flag on.

`$obj->`**open**(%options)
Open the folder which is described by this identity. Returned is some Mail::Box. The options are passed to **Mail::Box::Manager::open()**.

`$obj->`**remove**([$name])
Remove the folder (plus subfolders) with the `$name`. Without `$name`, this `Mail::Box::Identity` itself is removed.

The removed structure is returned, which is `undef` if not found. This is only an administrative remove, you still need a **Mail::Box::Manager::delete()**.

`$obj->`**rename**(`$folder`, [$newsubname])
Move the folder to a different super-$folder, under a NEW SUBfolder NAME.

example: renaming a folder

```
my $top = $user->topfolder;
my $new = $top->folder('xyz') or die;
my $f   = $top->folder('abc', 'def')->rename($new, '123');

print $f->name;      # 123
print $f->fullname;  # =/xyz/123
```

`$obj->`**subfolderNames**()
Convenience method: returns the names of the collected subfolders.

`$obj->`**subfolders**()
Returns the subfolders or `undef` if there are none. This information is lazy evaluated and cached. In LIST context, the folder objects are returned (Mail::Box::Identity objects), in SCALAR context the collection, the Mail::Box::Collection.

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::Identity->**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, [$strings]])
Mail::Box::Identity->**log**([$level, [$strings]])
Inherited, see ''Error handling'' in Mail::Reporter

`$obj->`**logPriority**($level)
Mail::Box::Identity->**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

DIAGNOSTICS

Error: `$object` is not a collection.
The first argument is an object, but not of a class which extends User::Identity::Collection.

Error: Cannot load collection module for `$type` ($class).
Either the specified `$type` does not exist, or that module named `$class` returns compilation errors. If the type as specified in the warning is not the name of a package, you specified a nickname which was not defined. Maybe you forgot the ’require’ the package which defines the nickname.

Error: Creation of a collection via `$class` failed.
The `$class` did compile, but it was not possible to create an object of that class using the options you specified.

Error: Don’t know what type of collection you want to add.
If you add a collection, it must either by a collection object or a list of options which can be used to create a collection object. In the latter case, the type of collection must be specified.

Error: It is not permitted to add subfolders to `$name`
The `$m`<**inferiors()**> flag prohibits the creation of subfolders to this folder.

Warning: No collection `$name`
The collection with `$name` does not exist and can not be created.

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.

Error: The toplevel folder cannot be removed this way
The Mail::Box::Identity folder administration structure requires a top directory. That top is registered somewhere (for instance by a Mail::Box::Manage::User). If you need to remove the top, you have to look for a method of that object.

Error: Toplevel directory requires explicit folder type
Error: Toplevel directory requires explicit location

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