NAME

User::Identity::Collection − base class for collecting roles of a user

INHERITANCE

```
 User::Identity::Collection
   is a User::Identity::Item

 User::Identity::Collection is extended by
   User::Identity::Collection::Emails
   User::Identity::Collection::Locations
   User::Identity::Collection::Systems
   User::Identity::Collection::Users
```

SYNOPSIS

```
 use User::Identity;
 use User::Identity::Collection;
 my $me    = User::Identity−>new(...);
 my $set   = User::Identity::Collection::Emails−>new(...);
 $me−>addCollection($set);

 # Simpler
 use User::Identity;
 my $me    = User::Identity−>new(...);
 my $set   = $me−>addCollection(type => 'email', ...)
 my $set   = $me−>addCollection('email', ...)

 my @roles = $me−>collection('email');  # list of collected items

 my $coll  = $me−>collection('email');  # a User::Identity::Collection
 my @roles = $coll−>roles;
 my @roles = @$coll;                    # same, by overloading

 my $role  = $me−>collection('email')−>find($coderef);
 my $role  = $me−>collection('location')−>find('work');
 my $role  = $me−>find(location => 'work');
```

DESCRIPTION

The `User::Identity::Collection` object maintains a set user related objects. It helps selecting these objects, which is partially common to all collections (for instance, each object has a name so you can search on names), and sometimes specific to the extension of this collection.

Currently imlemented extensions are

- *people* is a collection of users
- *whereabouts* are locations
- a *mailinglist* is a

  collection of email addresses
- a *network* contains

  groups of systems

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

OVERLOADED

overload: **@{}**

When the reference to a collection object is used as array-reference, it will be shown as list of roles.

example:

```
 my $locations = $ui->collection('location');
 foreach my $loc (@$location) ...
 print $location->[0];
```

overload: **stringification**

Returns the name of the collection and a sorted list of defined items.

example:

```
 print "$collection\n";  #   location: home, work
```

METHODS

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

Constructors

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

User::Identity::Collection->**new**([$name], %options)

```
 -Option     --Defined in          --Default
  description  User::Identity::Item  undef
  item_type                           <required>
  name         User::Identity::Item  <required>
  parent       User::Identity::Item  undef
  roles                               undef
```

description => STRING

item_type => CLASS

The CLASS which is used to store the information for each of the maintained objects within this collection.

name => STRING

parent => OBJECT

roles => ROLE|ARRAY

Immediately add some roles to this collection. In case of an ARRAY, each element of the array is passed separately to **addRole()**. So, you may end-up with an ARRAY of arrays each grouping a set of options to create a role.

Attributes

Extends "Attributes" in User::Identity::Item.

$obj->**description**()

Inherited, see "Attributes" in User::Identity::Item

$obj->**itemType**()

Returns the type of the items collected.

$obj->**name**([$newname])

Inherited, see "Attributes" in User::Identity::Item

$obj->**roles**()

Returns all defined roles within this collection. Be warned: the rules are returned in random (hash) order.

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**()
User::Identity::Collection−>**type**()
Inherited, see "Collections" in User::Identity::Item

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

Maintaining roles

`$obj->`**addRole**($role| <[$name],%options> | ARRAY)
Adds a new role to this collection. `$role` is an object of the right type (depends on the e xtension of this module which type that is) or a list of `%options` which are used to create such role. The options can also be passed as reference to an ARRAY. The added role is returned.

example:

```
my $uicl = User::Identity::Collection::Locations->new;

my $uil  = User::Identity::Location->new(home => ...);
$uicl->addRole($uil);

$uicl->addRole( home => address => 'street 32' );
$uicl->addRole( [home => address => 'street 32'] );
```

Easier

```
$ui      = User::Identity;
$ui->add(location => 'home', address => 'street 32' );
$ui->add(location => [ 'home', address => 'street 32' ] );
```

`$obj->`**removeRole**($role|$name)
The deleted role is returned (if it existed).

`$obj->`**renameRole**(<$role|$oldname>, `$newname`)
Give the role a different name, and move it in the collection.

`$obj->`**sorted**()
Returns the roles sorted by name, alphabetically and case-sensitive.

Searching

Extends "Searching" in User::Identity::Item.

`$obj->`**find**($name|CODE|undef)
Find the object with the specified `$name` in this collection. With `undef`, a randomly selected role is returned.

When a code reference is specified, all collected roles are scanned one after the other (in unknown order). For each role,

```
CODE->($object, $collection)
```

is called. When the CODE returns true, the role is selected. In list context, all selected roles are returned. In scalar context, the first match is returned and the scan is aborted immediately.

example:

```
my $emails = $ui->collection('emails');
$emails->find('work');

sub find_work($$) {
   my ($mail, $emails) = @_;
   $mail->location->name eq 'work';
}
my @at_work = $emails->find(\&find_work);
my @at_work = $ui->find(location => \&find_work);
my $any     = $ui->find(location => undef );
```

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 create a `$type` to add this to my collection.
: Some options are specified to create a `$type` object, which is native to this collection. However, for some reason this failed.

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: Cannot rename `$name` into `$newname:` already exists
Error: Cannot rename `$name` into `$newname:` doesn't exist
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.

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

Error: Wrong type of role for `$collection:` requires a `$expect` but got a `$type`
: Each `$collection` groups sets of roles of one specific type ($expect). You cannot add objects of a different `$type`.

SEE ALSO

This module is part of User-Identity distribution version 1.01, built on February 11, 2022. Website: *http://perl.overmeer.net/CPAN/*

LICENSE