NAME

User::Identity − maintain info about a physical person

INHERITANCE

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

SYNOPSIS

```
 use User::Identity;
 my $me = User::Identity−>new
  ( 'john'
  , firstname => 'John'
  , surname   => 'Doe'
  );
 print $me−>fullName  # prints "John Doe"
 print $me;           # same
```

DESCRIPTION

The `User−Identity` distribution is created to maintain a set of informational objects which are related to one user. The`User::Identity` module tries to be smart pro viding defaults, conversions and often required combinations.

The identities are not implementing any kind of storage, and can therefore be created by any simple or complex Perl program. This way, it is more flexible than an XML file to store the data. For instance, you can decide to store the data with Data::Dumper, Storable, DBI, AddressBook or whatever. Extension to simplify this task are still to be developed.

If you need more kinds of user information, then please contact the module author.

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

OVERLOADED

`$obj`−>**stringification**()

When an `User::Identity` is used as string, it is automatically translated into the **fullName()** of the user involved.

example:

```
 my $me = User::Identity−>new(...)
 print $me;          # same as  print $me−>fullName
 print "I am $me\n"; # also stringification
```

METHODS

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

Constructors

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

User::Identity−>**new**([$name], `%options`)

Create a new user identity, which will contain all data related to a single physical human being. Most user data can only be specified at object construction, because they should never change. A `$name` may be specified as first argument, but also as option, one way or the other is required.

```
 -Option     --Defined in          --Default
  birth                              undef
  charset                            $ENV{LC_CTYPE}
  courtesy                           undef
  description  User::Identity::Item  undef
  firstname                          undef
  formal_name                        undef
  full_name                          undef
  gender                             undef
  initials                           undef
  language                           'en'
  name         User::Identity::Item  <required>
  nickname                           undef
  parent       User::Identity::Item  undef
  prefix                             undef
  surname                            undef
  titles                             undef
```

birth => DATE
charset => STRING
courtesy => STRING
description => STRING
firstname => STRING
formal_name => STRING
full_name => STRING
gender => STRING
initials => STRING
language => STRING
name => STRING
nickname => STRING
parent => OBJECT
prefix => STRING
surname => STRING
titles => STRING

Attributes

Extends “Attributes” in User::Identity::Item.

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

Calcuted from the datge of birth to the current moment, as integer. On the birthday, the number is incremented already.

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

Returns the date in standardized format: YYYYMMDD, easy to sort and select. This may return `undef`, even if the **dateOfBirth()** contains a value, simply because the format is not understood. Month or day may contain `'00'` to indicate that those values are not known.

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

The user’s preferred character set, which defaults to the value of LC_CTYPE environment variable.

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

The courtesy is used to address people in a very formal way. Values are like “Mr.”, “Mrs.”, “Sir”, “Frau”, “Heer”, “de heer”, “mevrouw”. This often provides a way to find the gender of someone addressed.

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

Returns the date of birth, as specified during instantiation.

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

`$obj->`**firstname**()
: Returns the first name of the user. If it is not defined explicitly, it is derived from the nickname, and than capitalized if needed.

`$obj->`**formalName**()
: Returns a formal name for the user. If not defined as instantiation parameter (see**new()**), it is constructed from other available information, which may result in an incorrect or an incomplete name. The result is built from "courtesy initials prefix surname title".

`$obj->`**fullName**()
: If this is not specified as value during object construction, it is guessed based on other known values like "firstname prefix surname". If a surname is provided without firstname, the nickname is taken as firstname. When a firstname is provided without surname, the nickname is taken as surname. If both are not provided, then the nickname is used as fullname.

`$obj->`**gender**()
: Returns the specified gender of the person, as specified during instantiation, which could be like 'Male', 'm', 'homme', 'man'. There is no smart behavior on this: the exact specified value is returned. Methods **isMale()**, **isFemale()**, and **courtesy()** are smart.

`$obj->`**initials**()
: The initials, which may be derived from the first letters of the firstname.

`$obj->`**isFemale**()
: See **isMale()**: return true if we are sure the user is a woman.

`$obj->`**isMale**()
: Returns true if we are sure that the user is male. This is specified as gender at instantiation, or derived from the courtesy value. Methods isMale and isFemale are not complementatory: they can both return false for the same user, in which case the gender is undertermined.

`$obj->`**language**()
: Can contain a list or a single language name, as defined by the RFC Examples are 'en', 'en−GB', 'nl−BE'. The default language is 'en' (English).

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

`$obj->`**nickname**()
: Returns the user's nickname, which could be used as username, e−mail alias, or such. When no nickname was explicitly specified, the name is used.

`$obj->`**prefix**()
: The words which are between the firstname (or initials) and the surname.

`$obj->`**surname**()
: Returns the surname of person, or `undef` if that is not known.

`$obj->`**titles**()
: The titles, degrees in education or of other kind. If these are complex, you may need to specify the formal name of the users as well, because smart formatting probably failes.

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−>**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

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.

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

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

Copyrights 2003−2022 by [Mark Overmeer <markov@cpan.org>]. For other contributors see ChangeLog.

This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself. See *http://dev.perl.org/licenses/*