python-suseapi Documentation

Release 0.25

Michal Čihař

Aug 01, 2019

Contents

Contents:

CHAPTER 1

Installation

You can download sources at GitHub:

https://github.com/openSUSE/python-suseapi/

Package is also on PyPi:

https://pypi.python.org/pypi/python-suseapi

You can install it using pip:

```
pip install python-suseapi
```

CHAPTER 2

SUSEAPI command line interface

2.1 Synopsis

```
suseapi <command> [options]
```

Commands actually indicate which operation should be performed.

2.2 Description

This module also installs **suseapi** program, which allows you to easily access some of the functionality from command line. Currently following subcommands are available:

lookup-user `[--by BY] [--attribs ATTRIBS] value`
Lookups user information using *suseapi.userinfo*.

absence `value`
Lookups user absence information using *suseapi.presence*.

2.3 Files

~/.config/suseapi User configuration file

/etc/xdg/suseapi Global configration file

The program follows XDG specification, so you can adjust placement of config files by environment variables `XDG_CONFIG_HOME` or `XDG_CONFIG_DIRS`.

The configuration file is INI file, for example:

```
[ldap]
server = ldap://pan.suse.de
base = o=Novell

[presence]
servers = present.suse.de,bolzano.suse.de/nosend
```

2.4 Examples

Listing absences for user mcihar:

```
$ suseapi absence mcihar
2015-04-06 - 2015-04-06
2015-05-01 - 2015-05-01
2015-05-08 - 2015-05-08
2015-07-06 - 2015-07-06
2015-09-28 - 2015-09-28
2015-10-28 - 2015-10-28
```

Listing LDAP attributes for user mcihar:

```
$ suseapi lookup-user --attribs COMPANY,FULLNAME,uid mcihar
[('cn=MCihar,o=Novell',
  {'COMPANY': ['SUSE'], 'FULLNAME': ['Michal Cihar'], 'uid': ['mcihar']})]
```

CHAPTER 3

Examples

3.1 Bugzilla

Getting single bug from bugzilla:

```
from suseapi.bugzilla import Bugzilla
bugzilla = Bugzilla('user', 'pass')
bug = bugzilla.get_bug(123456)
```

Searching for bugs changed in last hour:

```
from suseapi.bugzilla import Bugzilla
bugzilla = Bugzilla('user', 'pass')
bugs = bugzilla.bugzilla.do_search([
    ('chfieldfrom', '1h'),
])
```

CHAPTER 4

API documentation

4.1 suseapi.browser

This module wraps `mechanize` module to provide higher level of abstraction for our needs.

exception `suseapi.browser.`**WebScraperError**

Base class for all web scaper errors.

class `suseapi.browser.`**WebScraper**(*user*, *password*, *base*, *useragent=None*)

request(*action*, *paramlist=None*, ***kwargs*)

Performs single request.

set_cookies(*cookies*)

Parameters **cookies** (*List of strings*) – Cookies to set

Sets authentication cookies.

get_cookies()

Returns Authentication cookies

Return type List of strings

Gets list of authentication cookies.

4.2 suseapi.bugzilla

This module allows remote access to Bugzilla. It wraps XML interface to read Bugzilla and SOAP service for writing to Bugzilla.

exception `suseapi.bugzilla.`**BugzillaError**

Base class for all Bugzilla errors.

exception suseapi.bugzilla.**BugzillaNotPermitted**
Operation was not permitted by Bugzilla.

exception suseapi.bugzilla.**BugzillaNotFound**
Bug was not found.

exception suseapi.bugzilla.**BugzillaInvalidBugId**
Bug ID is invalid.

exception suseapi.bugzilla.**BugzillaConnectionError**
Failed to connect to bugzilla.

exception suseapi.bugzilla.**BugzillaLoginFailed**
Login failed.

exception suseapi.bugzilla.**BuglistTooLarge**
The search result is too long.

exception suseapi.bugzilla.**BugzillaUpdateError**
Error while updating bugzilla field.

class suseapi.bugzilla.**Bug**(*bug_et*, *anonymous=False*)

Parameters **bug_et** (*ElementTree instance*) – Data obtained from XML interface

This class holds all data for single bug from Bugzilla. All XML elements are parsed to the Bug class attributes, so you can access them like `bug.bug_severity`.

class suseapi.bugzilla.**Bugzilla**(*user*, *password*, *base='https://bugzilla.novell.com'*)

Parameters

- **user** (*string*) – Username to Bugzilla
- **password** (*string*) – Password to Bugzilla
- **base** (*string*) – Base URL for Bugzilla

Bugzilla communication class for read only access. With iChain authentication. The authentication part is expensive so it is good idea to remember authentication cookies and reuse them as much as possible. It is subclass of *suseapi.browser.WebScraper*.

login()

Throws *BugzillaLoginFailed* in case login fails.

Performs login to Bugzilla.

get_bug(*bugid*, *retry=True*)

Parameters

- **bugid** (*integer*) – Bug id
- **retry** (*boolean*) – Whether to retry with new login on failure

Returns Bug data

Return type *Bug* instance

Reads single bug from Bugzilla.

get_bugs(*ids*, *retry=True*, *permissive=False*, *store_errors=False*)

Parameters

- **ids** (*list of integers*) – Bug ids

- **retry** (*boolean*) – Whether to retry with new login on failure
- **permissive** (*boolean*) – Whether to ignore not found bugs
- **store_errors** (*boolean*) – Whether to store bug retrieval errors in result

Returns Bug data

Return type list of *Bug* instances

Reads list of bugs from Bugzilla.

do_search(params):

Parameters **params** (*list of tuples*) – URL parameters for search

Returns List of bug ids

Return type list of integers

Throw *BuglistTooLarge* in case search result is too long.

Searches for bugs matching given criteria, you can construct the query based on the bugzilla web interface.

get_recent_bugs(*startdate*)

Parameters **startdate** (*datetime instance*) – Date from which to search.

Returns List of bug ids

Return type list of integers

Throw *BuglistTooLarge* in case search result is too long.

Gets list of bugs modified since defined date.

get_openl3_bugs()

Returns List of bug ids

Return type list of integers

Throw *BuglistTooLarge* in case search result is too long.

Searches for bugs with openL3 in whiteboard.

get_l3_summary_bugs()

Returns List of bug ids

Return type list of integers

Throw *BuglistTooLarge* in case search result is too long.

Searches for open bugs with L3: in summary.

get_sr(*bugid*)

Parameters **bugid** (*integer*) – Bug id

Return type list of integers

Returns list of SRs associated with given bug.

update_bug(*bugid*, *callback=None*, *callback_param=None*, *whiteboard_add=None*, *whiteboard_remove=None*, ***kwargs*)

Parameters **bugid** (*integer*) – Bug id

Updates single bug in bugzilla.

class suseapi.bugzilla.**APIBugzilla**(*user*, *password*, *base='https://apibugzilla.novell.com'*)
Wrapper around `suseapi.bugzilla.Bugzilla` class to use HTTP authentication instead of iChain.

class suseapi.bugzilla.**DjangoBugzilla**(*user*, *password*, *base='https://apibugzilla.novell.com'*)
Wrapper around `suseapi.bugzilla.APIBugzilla` class to use Django logging.

suseapi.bugzilla.**get_django_bugzilla**()

Return type object

Returns DjangoBugzilla instance

Constructs `DjangoBugzilla` objects with cookie persistence in Django cache, so the there is no need to login on every request.

4.3 suseapi.presence

class suseapi.presence.**Presence**(*hosts=None*)

Parameters **hosts** (*list*) – List of hosts to query

Class for querying (and caching) presence data. The optional hosts list can define which hosts will be used for querying presence database.

get_presence_data(*person*)

Parameters **person** (*string*) – Username

Return type list

Returns List of absences

Returns list of absences for given person.

is_absent(person, when, threshold=0):

Parameters

- **person** (*string*) – Username
- **when** (*date*) – Date
- **threshold** (*integer*) – Threshold for presence check

Return type bool

Checks whether person is absent on given date.

The optional threshold parameter can specify how long absences to ignore. For example setting it to 1 will ignore one day absences which would otherwise make the method return true.

4.4 suseapi.srinfo

This module allows remote access to SR database.

class suseapi.srinfo.**SRInfo**

get_status(*srid*)

Parameters **srid** (*integer*) – SR id

Return type string

Returns String with status

Returns SR status.

get_info(*srid*)

Parameters **srid** (*integer*) – SR id

Return type dict

Returns Dictionary with SR attributes

Returns SR status.

class suseapi.srinfo.**DjangoSRInfo**

Wrapper around *suseapi.srinfo.SRInfo* class to use Django settings and cache results in Django cache.

4.5 suseapi.swamp

This module allows remote access to SWAMP service. It is basically just a wrapper around suds to workaround some weirdness which SWAMP SOAP interface exposes.

class suseapi.swamp.**SWAMP**(*user*, *password*, *url=None*, *tracefile=None*)

Parameters

- **user** (*string*) – User name.
- **password** (*string*) – Password to authenticate to SWAMP.
- **url** (*string*) – SWAMP URL (default is http://swamp.suse.de:8080/axis/services/swamp)
- **tracefile** (*file object*) – File handle where SOAP traces will be written.

getMethodDoc(*name*)

Gets online documentation for method.

Parameters **name** (*string*) – Name of method

Returns Documentation for method

Return type string

getAllDocs()

Gets online documentation for all methods.

Returns Documentation for all methods

Return type dict

login()

Logins to SWAMP.

This actually only tests whether login information is correct.

Returns None

doGetProperty(*name*)

Gets SWAMP property.

Parameters **name** (*string*) – Name of property

Returns Value of property

Return type string

getWorkflowInfo(*id*)
Gets the workflows properties.

Parameters **id** (*integer*) – Workflow ID.

Returns Workflow properties.

doGetAllDataPaths(*id*)
Gets all workflows data paths.

Parameters **id** (*integer*) – Workflow ID.

Returns Workflow data paths.

doGetData(*id*, *path*)
Gets workflow data bit.

Parameters

- **id** (*integer*) – Workflow ID.
- **path** (*string*) – Data path.

Returns Workflow data bit value.

doGetAllData(*id*)
Gets all workflow data bits.

Parameters **id** (*integer*) – Workflow ID.

Returns Workflow data bit values.

Return type dict

getDataBit(*id*, *path*)
Efficient wrapper around *doGetAllData()* and *doGetData()* to get a data bit. It first tries to use all data, because getting it takes same time as single bit, but the data is cached and reused for next time.

Parameters

- **id** (*integer*) – Workflow ID.
- **path** (*string*) – Data path.

Returns Workflow data bit value.

Return type string

doSendData(*id*, *path*, *value*)
Sets data bit in a workflow.

Parameters

- **id** (*integer*) – Workflow ID.
- **path** (*string*) – Data path.
- **value** (*string*) – Data value.

Returns None

doSendEvent(*id*, *envent*)
Sets data bit in a workflow.

Parameters

- **id** (*integer*) – Workflow ID.
- **event** (*string*) – Event name.

Returns None

4.6 `suseapi.userinfo`

This module allows remote access to LDAP. It wraps standard Python module for LDAP and provides some convenience functions.

class suseapi.userinfo.**UserInfo**(*server*, *base*)

Parameters

- **server** (*string*) – Server address
- **base** (*string*) – Search base

LDAP class wrapping ldap access.

search_uid(*uid*, *attribs=None*)

Parameters

- **uid** (*string*) – Search string
- **attribs** (*list of strings*) – Attributes to read from LDAP, defaults to [‘cn’, ‘mail’, ‘ou’, ‘sn’, ‘givenName’]

Return type list of dictionaries

Returns Search results

Performs UID search and returns list of search results.

search_by(*attr*, *val*, *attribs=None*)

Parameters

- **attr** – attribute name to search by
- **val** – value of the attribute to search for
- **attribs** – attributes to return

Performs search by any attribute.

get_department(*user*)

Parameters **user** (*string*) – Search string

Return type string

Returns Department name, `N/A` in case it was not found.

Performs LDAP search and grabs department name from it. Additionally some fixups are applied to department names to avoid more names for single department.

class suseapi.userinfo.**DjangoUserInfo**(*server*, *base*)

Wrapper around *suseapi.userinfo.UserInfo* class to use Django settings and cache results in Django cache.

CHAPTER 5

Developing

5.1 Testsuite

The testsuite can be executed using `py.test`.

5.2 Continuous integration

We're using several services to ensure our code is healthy:

Travis CI for running the testsuite https://travis-ci.org/openSUSE/python-suseapi

Coveralls for reporting the testsuite coverage https://coveralls.io/r/openSUSE/python-suseapi?branch=master

Landscape for checking quality of our code https://landscape.io/github/openSUSE/python-suseapi/master

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index