Dr.Web pyavdesk Documentation

Release 1.1.0

Doctor Web, Ltd.

January 29, 2014

Contents

Intoduction

Dr.Web pyavdesk is a Python module, wrapping Dr.Web *dwavdapi* C library to introduce simple yet powerful interface to manipulatate Dr.Web AV-Desk server resources from within your Python code, making integration of thirdparty applications with AV-Desk as easy as possible.

CHAPTER 1

Requirements

- Python 2.6+
- Dr.Web AV-Desk (http://ru.av-desk.com/support/demo/?lng=en)
- Dr.Web *dwavdapi* library (http://download.geo.drweb.com/pub/drweb/avd/libs/libdwavdapi/)
- Python setuptools (to install pyavdesk package using setup.py)

CHAPTER 2

Note on examples

If you are using Python 3 please take into account that all code examples in this documentation are for Python 2 series syntax.

Contents

3.1 Getting started with pyavdesk

Before using pyavdesk please verify that Dr.Web *dwavdapi* library is installed and available in your system. Library installation process is beyond the scope of this documentation.

Note: On Microsoft Windows systems *dwavdapi* library should reside in folder with system DLLs, e.g. *<DRIVE_LETTER>:\Windows\system32\* for NT family systems.

3.1.1 Resource classes

pyavdesk interacts with AV-Desk servers by means of several classes, each of which describes a certain server resource.

- Server operations are available from *AVDeskServer* class.
- Administrators are created and modified with the help of *AVDeskAdmin* class.
- Tools for groups manipulations are available at *AVDeskGroup* class.
- It is possible to manage server tariffs with *AVDeskTariff* class.
- At last *AVDeskStation* class introduces an interface for station control.

3.1.2 Quick example

AVDeskServer class documentation shall be a good starting point, yet below is an example on how to connect to server and to fetch and create some resources:

```
from pyavdesk import pyavdesk

# Check that dwavdapi library version suits pyavdesk.
lib_satisfied, lib_version = pyavdesk.library_version_satisfied()
if not lib_satisfied:
    print 'WARNING: System has a mismatching version of dwavdapi library - %s' % lib_version

# Setting up server connection parameters - login, password and URL.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')
# Use only trusted connections.
av_server.verify_connection_certificate(True)

administrators = []
```

```
try:
    # Get a list of administrators registered on server.
    administrators = av_server.get_administrators()
except pyavdesk.AVDeskError as e:
    print ’Unable to fetch administrators from server. Rough reason: %s’ % e.message

print ’Server has %s administrator accounts.’ % len(administrators)

# Now if there is no administrator with name ‘my_demo_admin‘ on server we’ll create one.
if ’my_demo_admin’ not in [administrator.login for administrator in administrators]:

    # Creating administrator object and setting its parameters.
    demo_admin = av_server.new_administrator(’my_demo_admin’, ’my_demo_password’)
    demo_admin.name = ’Fred’
    demo_admin.last_name = ’Colon’

    # Creating group object for future administrator to watch for.
    demo_group = av_server.new_group(’Colon\’s Group’)
    demo_group.description = ’This is a group lead by Fred Colon.’
    demo_group.add_emails([’fred@colon.dcw’])

    try:
        # Trying to save group.
        demo_group.save()
    except pyavdesk.AVDeskError as e:
        print ’Unable to create Fred\’s group. Reason: %s’ % e.message
    else:
        # After the group has been created we link it to out administrator.
        demo_admin.add_to_groups([demo_group])

    try:
        # Trying to save administrator object.
        demo_admin.save()
    except pyavdesk.AVDeskError as e:
        print ’Unable to create Fred\’s administrator account. Reason: %s’ % e.message
    else:
        print ’New administrator named %s is created. His ID is "%s". Global admin - %s. Subjected gr
              (demo_admin.name, demo_admin.id, demo_admin.is_global_admin, demo_admin.get_groups())

    try:
        # Cleaning up.
        demo_admin.delete()
        demo_group.delete()
    except pyavdesk.AVDeskError:
        print ’ERROR: Unable to delete demo administrator and his group from server.’
    else:
        print ’Demo administrator and his group have been deleted from server.’
```

3.2 Handling errors

pyavdesk ships with three exception classes (`AVDeskError`, `AVDeskLibraryError` and `AVDeskServerError`) to handle errors.

3.2.1 AVDeskError

This is a basic exception. All the other `AVDesk` family exceptions classes are inhereted from it.

Use it to handle any type of error connected with pyavdesk functionality.

Exception properties:

- `message` — Error description returned by *dwavdapi* library or generated by pavdesk.
- `lib_function` — dwavdapi library function name on which execution error has occured.
- `lib_errno` — Error code returned by *dwavdapi* library.
- `lib_function_args` — Arguments passed to *dwavdapi* library function.

Example:

```
# Trying to get non-existing station by its ID leads to AVDeskError exception also.
try:
    station = av_server.get_station(’non_existing_station_id’)
except pyavdesk.AVDeskError as e:
    print ’ERROR at ‘%s‘: %s. Code:  %s. Description: %s.’ % (e.lib_function, e.lib_errno, e)
```

3.2.2 AVDeskLibraryError

This exception class is inhereted from `AVDeskError` and is raised when pyavdesk fails to communicate with *dwadvapi* library or there is an exception in library itself.

Use it to handle pure library errors.

Example:

```
# Handling hypothetical library error while getting ‘Everyone‘ group from server.
try:
    everyone = av_server.get_group(pyavdesk.META_GROUP_IDS[’EVERYONE’])
except AVDeskLibraryError:
    print ’ERROR at ‘%s‘. Code: %s. Description: %s.’ % (e.lib_function, e.lib_errno, e)
```

3.2.3 AVDeskServerError

This exception class is inhereted from `AVDeskError` and is raised when *dwadvapi* library recieved an error response from server.

Use it to handle server errors.

Example:

```
# Trying to get non-existing tariff by its ID leads to AVDeskServerError exception.
try:
    subtariff = av_server.get_tariff(’non_existing_tariff_id’)
except pyavdesk.AVDeskServerError as e:
    print e
```

3.3 Utilities

pyavdesk comes with some utility constants and functions that could be of use.

3.3.1 META_GROUP_IDS

META_GROUP_IDS dictionary contains identifiers for basic groups on every AV-Desk server.

Usage example:

```
everyone = av_server.get_group(pyavdesk.META_GROUP_IDS['EVERYONE'])
```

3.3.2 TARIFF_IDS

TARIFF_IDS dictionary contains identifiers for basic tariff groups on every AV-Desk server.

Usage example:

```
tariff_classic = av_server.get_tariff(pyavdesk.TARIFF_IDS['CLASSIC'])
```

3.3.3 library_version_satisfied()

`library_version_satisfied()` function tests whether version of *dwavdapi* library found in the system is the same as version pyavdesk was designed to work with. Returns *True* if version is satisfied, otherwise *False*.

3.3.4 set_log_level()

`set_log_level(lvl)` function sets module logging to a desired level and turns on log pretty formatting. `lvl` parameter should be logging level from `logging` module. E.g.: logging.DEBUG.

3.4 Server - AVDeskServer class

Server resource is a basic resource of pyavdesk. It gives an access both to server properties (statistics, key, etc.) and other resources manipulation means (creating and fetching server groups, tariffs, stations, etc.).

Note: It is advised to use server resource class **AVDeskServer** as a single entry point for every server resource manipulation instead of direct instantiation of other resources classes (AVDeskGroup, AVDeskStation, etc.).

Basic class usage example:

```
# Setting up server connection parameters.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')

# Printing out server OS.
info = av_server.get_info()
print 'OS used - %s' % info['os']

# Printing out names of tariffs available at server.
tariffs = av_server.get_tariffs()
for tariff in tariffs:
    print 'Tariff "%s" is available' % tariff.name

# Getting server groups and printing out their IDs.
groups = av_server.get_groups()
for group in groups:
    print 'A group with ID - %s' % group.id
```

```
# Creating a new group.
my_group = av_server.new_group('My Group')
my_group.save()
print 'Server generated group ID - "%s"' % my_group.id

# Getting server stations ID printed out.
# Note though that some AV-Desk versions may return an empty list here.
stations = av_server.get_stations()
for station in stations:
    print 'Station found - %s' % station
```

class `pyavdesk.pyavdesk.`**`AVDeskServer`**(*login*, *password*, *url='http://127.0.0.1'*, *port=9080*, *connection_timeout=2*)

This is a basic pyavdesk class, providing server connection handle, and essential methods to manipulate server resources.

Note: If `url` parameter value contains *https* dwavdapi library will try to verify connection certificate automatically. This behavior can be changed with `verify_connection_certificate()`.

Parameters

- **login** – is the login of user from whose name module issues connection
- **password** – user password used for connection
- **url** – should be set to AV-Desk web-server URL
- **port** – should be set to AV-Desk web-server port
- **connection_timeout** – connection timeout in seconds

Returns `AVDeskServer` instance

Raises `AVDeskError` on failure

Example:

```
# Setting up the connection data with timeout equal to 5 seconds.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70', connection_timeout
```

`get_administrator`(*administrator_login*)

Returns administrator resource object from server by login.

Parameters **administrator_login** – administrator login to retrieve information for

Returns `AVDeskAdmin` instance

Raises `AVDeskError` on failure

Example:

```
admin_fred = av_server.get_administrator('fred_colon')
print 'Admin ID - %s' % admin_fred.id
```

`get_administrators`()

Returns a list of administator objects registered at server, performing server call.

Note: Objects in the list returned have only basic information. To get full infomation use `AVDeskAdmin.retrieve_info()`.

Returns list of `AVDeskAdmin` instances

Raises `AVDeskError` on failure

Example:

```
administrators = av_server.get_administrators()
for administrator in administrators:
    print 'An administrator with login - %s' % administrator.login
```

`get_group`(*group_id*)

Returns group resource object from server by ID.

Parameters **group_id** – group ID to retrieve information for

Returns `AVDeskGroup` instance

Raises `AVDeskError` on failure

Example:

```
everyone = av_server.get_group(pyavdesk.META_GROUP_IDS['EVERYONE'])
```

In this example we use **META_GROUP_IDS** dictionary defined in the module to get 'EVERYONE' meta-group ID from to pass it to the method.

`get_groups`()

Returns a list of groups objects registered at server, performing server call.

Note: Objects in the list returned have only basic information. To get full infomation use `AVDeskGroup.retrieve_info()`.

Returns list of `AVDeskGroup` instances

Raises `AVDeskError` on failure

Example:

```
groups = av_server.get_groups()
for group in groups:
    print 'A group with ID - %s' % group.id
```

`get_info`()

Makes request to a server and returns a dictionary with basic information about server.

Example:

```
info = av_server.get_info()
```

An extract from dictionary returned:

```
{
    'api_build': '201110201',
    'groups_system': 85,
    'uptime': 98544,
    'tariffs_total': 25,
    'stations_total': 283,
    'os': 'Linux',
     ...
}
```

Returns dictionary

Raises AVDeskError on failure

get_key_info()

Makes request to a server and returns a dictionary with information about AV-Desk server key.

Example:

```
info = av_server.get_key_info()
```

An extract from dictionary returned:

```
{
    'expires_ts': 1341867133,
    'created_ts': 1310158333,
    'servers': 1,
    'sn': None,
    'md5': '6036761418df7065dab1518afd2cf1e7'
    ...
}
```

Returns dictionary

Raises AVDeskError on failure

get_repositories()

Makes request to a server and returns a dictionary with data about repositories used by server.

Returns list of dictionaries

Raises AVDeskError on failure

An extract from repositories data list:

```
[
    {
        'state': 'ok',
        'code': '20-drwwince',
        'name': 'Mobile Dr.Web AV-Desk Agent',
        'rev_date_ts': 1311239190,
        'rev_id': 1311239190
    },
    ...
]
```

Code sample:

```
repositories = av_server.get_repositories()
for repository in repositories:
    print 'Repository "%s" -> "%s"' % (repository['code'], repository['state'])
```

get_resource_id()

Helper method to get resource identifier, which can be passed to such resource manipulation methods as get_info() and delete().

Returns resource identifier as string

get_station(*station_id*)

Returns station resource object from server by ID.

Parameters **station_id** – station ID to retrieve information for

Returns `AVDeskStation` instance

Raises `AVDeskError` on failure

Example:

```
station = avdesk.get_station('my_station')
print station.name
```

get_stations(*as_id=True*)

Shortcut method that returns a list of stations registered at server through *Everyone* system group quering.

Parameters **as_id** – boolean flag. If *True* list of `AVDeskStation` instances is returned, if *False* - list of strings representing station IDs.

Returns list of objects which type is defined by *as_id* parameter

Raises `AVDeskError` on failure

> **Warning:** Setting **as_id** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full station information. One should bear it in mind when querying servers with large amount of stations.

Example:

```
stations = av_server.get_stations()
for station in stations:
    print 'Station ID - %s' % station.id
```

get_statistics(*ts_from=0*, *ts_till=0*, *virus_limit=10*)

Makes request to a server and returns a dictionary with overall server statistics.

Parameters

- **ts_from** – timestamp of when statistics starts. If 0 - statistics starts at year 1970. Default 0.
- **ts_till** – timestamp of when staristics ends. If 0 - statistics ends today. Default 0.
- **virus_limit** – virus statistics limiter, to return no more than a given number of viruses. Default 10.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
stats = av_server.get_statistics()
```

An extract from dictionary returned:

```
{
    'groups_system': 85,
    'tariffs_total': 25,
    'traffic': {
        'total': 1140022.0,
        'out': 322671.0,
        'in': 817351.0
    },
    'stations_total': 283,
    'stations_state': {
        'unactivated': 270,
        'activated': 13,
        'deinstalled': 0,
        ...
    },
    ...
}
```

get_tariff(*tariff_id*)

Returns tariff group resource object from server by ID.

Parameters **tariff_id** – tariff group ID to retrieve information for

Returns `AVDeskTariff` instance

Raises `AVDeskError` on failure

Example:

```
tariff_group = avdesk.get_tariff('my_tariff')
print tariff_group.name
```

get_tariffs()

Returns a list of tariff group objects registered at server, performing server call.

Note: Objects in the list returned have only basic information. To get full infomation use `AVDeskTariff.retrieve_info()`.

Returns list of `AVDeskTariff` instances

Raises `AVDeskError` on failure

Example:

```
tariffs = av_server.get_tariffs()
for tariff in tariffs:
    print 'Tariff named "%s"' % tariff.name
```

new_administrator(*administrator_login*, *password=None*)

Initializes new `AVDeskAdmin` object with given parameters and returns it for further usage.

Parameters

- **administrator_login** – login for new administrator
- **password** – password for new administrator. If *None* password is generated by server and available from `AVDeskAdmin.password` after object is saved.

Returns `AVDeskAdmin`

Example:

```
administrator = avdesk.new_administrator('fred_colon')
administrator.save()
print 'Server generated password - "%s"' % administrator.password
```

new_group(*group_name*, *id=None*, *parent_group=None*)

Initializes new `AVDeskGroup` object with given parameters and returns it for further usage.

Parameters

- **group_name** – name for the group

- **id** – group ID. If *None* group ID is generated by server and available from `AVDeskGroup.id` after object is saved
- **parent_group** – parent group ID or object. Note: parent group must already exist on server.

Returns `AVDeskGroup`

Example:

```
group = avdesk.new_group('My Group', parent_group=my_parent_group)
group.save()
print 'Server generated group ID - "%s"' % group.id
```

new_station(*id=None*, *parent_group=None*, *tariff=None*)

Initializes new `AVDeskStation` object with given parameters and returns it for further usage.

Parameters

- **id** – station ID. If *None* station ID is generated by server and available from `AVDeskStation.id` after object is saved.
- **parent_group** – parent group ID or object. Note: parent group must already exist on server.
- **tariff** – tariff group ID or object. Note: tariff group must already exist on server.

Returns `AVDeskStation`

Example:

```
station = avdesk.new_station('My Station', parent_group=my_parent_group, tariff='tariff_id')
station.save()
print 'Server generated station ID - "%s"' % station.id
```

new_tariff(*tariff_name*, *id=None*, *parent_tariff=None*, *grace_period=None*)

Initializes new `AVDeskTariff` object with given parameters and returns it for further usage.

Parameters

- **tariff_name** – name for the tariff group
- **id** – tariff group ID. If *None* tariff group ID is generated by server and available from `AVDeskTariff.id` after object is saved.
- **parent_tariff** – parent tariff group ID or object. Note: parent tariff group must already exist on server.
- **grace_period** – grace period rot this tariff in *days*.

Returns `AVDeskTariff`

Example:

```
tariff = avdesk.new_tariff('My Tariff', parent_tariff='my_parent_tariff_id')
tariff.save()
print 'Server generated tariff ID - "%s"' % tariff.id
```

run_task(*task_id*)

Requests a server to execute the task defined by its ID string.

Parameters **task_id** – task identifier string.

Raises `AVDeskError` on failure

set_user_agent(*title*)

Instructs *dwavdapi* library to use the given User Agent string on server requests.

Parameters **title** – User Agent string.

Returns boolean. *True* on success, overwise - *False*

Raises AVDeskError on failure

switch_to_debug_mode(*log_filepath=None*)

Permanently switches dwavdapi library into debug mode when debugging information including server responses is written into a file.

Parameters **log_filepath** – absolute path to file to write logs into. If *None libdwavdapi.log* in module's directory will be used.

Returns boolean. *True* if successfully switched, overwise - *False*

Raises AVDeskError on failure

verify_connection_certificate(*do_verify*, *certificate_path=None*)

Gives *dwavdapi* library an instruction to verify or not to verify connection certificate.

Parameters

- **do_verify** – boolean. If *True* connection certificate will be verified.
- **certificate_path** – expects full path to root certificate to verify connection validity against, or *None* is system certificates should be used.

Raises AVDeskError on failure

3.5 Administrators - AVDeskAdmin class

Administrator pyavdesk resource described by **AVDeskAdmin** class exposes an interface to manipulate server administrators.

Note: It is advised to use server resource class **AVDeskServer** as a single entry point for every server resource manipulation instead of direct instantiation of **AVDeskAdmin** class.

Basic class usage example:

```
# Setting up server connection parameters.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')

# Get main administrator (his login is 'admin').
superadmin = av_server.get_administrator('admin')
print superadmin.name

# Create new administrator.
new_admin = av_server.new_administrator('my_new_admin')
new_admin.name = 'Fred'
new_admin.last_name = 'Colon'
new_admin.save()
print 'Password for Fred is "%s"' % new_admin.password
```

class pyavdesk.pyavdesk.**AVDeskAdmin**(*connector_resource*, *resource_id=None*, *predefined_handle=None*)

AV-Desk administrator resource class is used to perform manipulations on administrators.

Parameters

- **connector_resource** – should be `AVDeskServer` instance.
- **resource_id** – if set server call is performed to get information for resource with given ID.
- **predefined_handle** – if set, resource data could be fetched from a resource at given handle.

Raises `AVDeskError` on failure

add_to_groups(*groups_list*)

Add groups to resource association by group IDs or objects.

Parameters **groups_list** – list of `AVDeskGroup` instances, or a list of strings representing groups IDs

Raises `AVDeskError` on failure

Note: Resource to group association is not sent to server until object's `save()` is not called.

Warning: Some AV-Desk server versions (namely 6.2) might handle only one group per administrator. In that case only the first group from the given list is linked with the administrator.

Example:

```
resource_obj.add_to_groups(['my_group_id', my_another_group_obj])
resource_obj.save()
```

create(*auto_retrieve=True*)

Performs a server call to create resource with properties defined in object.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *update* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
created = resource_obj.create()
```

delete(***kwargs*)

Performs server call in an attempt to delete the resource.

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
deleted = resource_obj.delete()
```

delete_from_groups(*groups_list*)

Removes groups to resource association by group IDs or objects.

Parameters groups_list – list of `AVDeskGroup` instances, or a list of strings representing groups IDs

Raises `AVDeskError` on failure

Note: Resource to group association is not sent to server until object's `save()` is not called.

Example:

```
resource_obj.delete_from_groups(['my_group_id', my_another_group_obj])
resource_obj.save()
```

get_groups(*as_id=True*)

Performs a server call and returns a list of groups associated with the resource.

Parameters as_id – boolean. If *True* list of `AVDeskGroup` instances is returned, if *False* - list of strings representing groups IDs

Returns list of objects which type is defined by *as_id* parameter

Raises `AVDeskError` on failure

Note: Objects in the list returned have only basic information. To get full infomation use `AVDeskGroup.retrieve_info()`.

Example:

```
groups = resource_obj.get_groups()
for group in groups:
    print 'Group ID - %s' % group.id
```

get_resource_id()

Helper method to get resource identifier, which can be passed to such resource manipulation methods as `get_info()` and `delete()`.

Returns resource identifier as string

is_global_admin

This *property* is a boolen used to identify administrator as global (i.e. his powers are not restricted to certain group(s)).

is_group_admin

This *property* is a boolen used to identify that administrator's powers are restricted to certain group(s).

last_name

This *property* is used to get or set administarator last name.

login

This *property* is used to get or set administarator login.

make_global_admin()

Shortcut method method that removes the administrator from all groups he is associated with, thus making him a global administrator.

Under the hood it utilizes `get_groups()` and `delete_from_groups()` methods.

Returns None

Raises `AVDeskError` on failure

Example:

```
my_admin.make_global_admin()
```

middle_name

This *property* is used to get or set administarator middle name.

password

This *property* is used to get or set administarator password.

readonly

This *property* is used to get or set administarator readonly boolean restriction flag. If readonly flag is set to True administrator would be unable to perform create/update/delete actions on server resources.

restrictions

This *property* is used to get or set restrictions for an administrator. Some operations on stations (e.g. changing expires, block period, parent group) will be restricted if True is passed. NOTE: This applies only to group administrators.

retrieve_info(*resource_id=None*)

Performs a server call to retrieve complete resource information by its ID and puts it into object's properties.

Parameters **resource_id** – specific ID of the resource. If *None*, ID is taken from the object itself.

Raises `AVDeskError` on failure

Example:

```
# resource_obj contains no additional info.
assert resource_obj.name is None

# After the following request
resource_obj.retrieve_info('some_resource_id')

# resource_obj contains additional info.
assert resource_obj.name is not None
```

right_create_admins

This *property* is used to get or set administarator's right to create other administrators. NOTE: Only for group administrators.

save(*auto_retrieve=True*)

A convinience method that automatically creates new resource on server if it doesn't exists or updates it if it does.

Under the hood it switches between `create()` and `update()` methods.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *save* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
saved = resource_obj.save()
```

update(*auto_retrieve=True*)

Performs server call in attempt to update the resource with information from class properties.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *update* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
updated = resource_obj.update()
```

3.6 Groups - AVDeskGroup class

pyavdesk group resource described by **AVDeskGroup** class exposes an interface to manipulate groups at server.

Note: It is advised to use server resource class **AVDeskServer** as a single entry point for every server resource manipulation instead of direct instantiation of **AVDeskGroup** class.

Basic class usage example:

```
# Setting up server connection parameters.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')

# Get `Everyone` group.
everyone = av_server.get_group(pyavdesk.META_GROUP_IDS['EVERYONE'])
print everyone.name

# Getting stations IDs from `Everyone`.
# Note though that some AV-Desk versions may return an empty list here.
stations = everyone.get_stations()
for id in stations:
    print 'Station ID - %s' % id

# Create a new group.
new_group = av_server.new_group('My group.')
new_group.save()
```

class pyavdesk.pyavdesk.**AVDeskGroup**(*connector_resource*, *resource_id=None*, *predefined_handle=None*)

AV-Desk group resource class is used to perform groups manipulations.

Parameters

- **connector_resource** – should be `AVDeskServer` instance.

- **resource_id** – if set server call is performed to get information for resource with given ID.
- **predefined_handle** – if set, resource data could be fetched from a resource at given handle.

Raises `AVDeskError` on failure

`add_emails`(*emails_list*)

Adds given emails to a list of e-mails defined for the resource.

Note: This method does not send data to server, for this to be done one needs to save resource object.

Parameters **emails_list** – list of e-mails

Example:

```
resource_obj.add_emails(['person1@server.com', 'person2@server.com'])
```

`add_station`(*station_obj*)

Adds a given station to current group making a server call. Creates current group on server if it doesn't already exist.

Parameters **station_obj** – `AVDeskStation` instance

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
my_station = av_server.new_station('My station')
saved = group.add_station(my_station)
```

`create`(*auto_retrieve=True*)

Performs a server call to create resource with properties defined in object.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *create* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
created = resource_obj.create()
```

`delete`(***kwargs*)

Performs server call in an attempt to delete the resource.

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
deleted = resource_obj.delete()
```

delete_emails(*emails_list*)

Removes given emails from a list of e-mails defined for the resource.

Note: This method does not send data to server, for this to be done one needs to save resource object.

Parameters emails_list – list of e-mails

Example:

```
resource_obj.delete_emails([’person1@server.com’, ’person2@server.com’, ’person3@server.com’
```

get_administrators(*as_logins=True*)

Returns a list of administators bound to a group, performing a server call.

Parameters as_logins – boolean. If *True* method returns a list of logins. If *False* - a list of `AVDeskAdmin` objects.

Returns list

Raises `AVDeskError` on failure

Warning: Setting **as_logins** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full administrator information. One should bear it in mind when querying groups with large amount of administrators.

Example:

```
administrators = resource_obj.get_administrators()
for login in administrators:
    print ’An administrator with login - %s’ % login
```

get_av_components()

Performs a server call to retrieve a list of antivirus application components information for the resource.

Returns list of dictionaries with components data

Raises `AVDeskError` on failure

Example:

```
components = resource_obj.get_av_components()
```

An extract from returned list:

```
[
    {
        ’status’: 1,
        ’code’: 105,
        ’parent_group_id’: ’2888b7ff-3625-465e-bcb8-957de17f6458’,
        ’code_text’: ’Dr.Web Firewall’,
        ’status_text’: ’Optional’,
        ...
    },
    ...
]
```

get_av_rights()

Performs a server call to retrieve a list of rights defined for the resource.

Returns list of dictionaries with rigths data

Raises `AVDeskError` on failure

Example:

```
rights = resource_obj.get_av_rights()
```

An extract from returned list:

```
[
    {
        'status': 1,
        'code': 53,
        'parent_group_id': '2888b7ff-3625-465e-bcb8-957de17f6458',
        'code_text': 'Uninstall Dr.Web Agent',
        'status_text': 'Enabled',
        ...
    },
    ...
]
```

get_emails()

Returns a list of e-mails defined for the resource.

Returns list of e-mails

Example:

```
emails = resource_obj.get_emails()
```

get_key()

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

get_parent(*as_id=True*)

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print 'Parent ID - %s' % parent
```

get_resource_id()

Helper method to get resource identifier, which can be passed to such resource manipulation methods as `get_info()` and `delete()`.

Returns resource identifier as string

get_stations(*as_id=True*)

Returns a list of stations in a group, performing a server call.

Parameters **as_id** – boolean. If *True* method returns a list of stations IDs. If *False* - a list of `AVDeskStation` objects.

Returns list

Raises `AVDeskError` on failure

> **Warning:** Setting **as_id** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full station information. One should bear it in mind when querying groups with large amount of stations.

Example:

```
stations = resource_obj.get_stations()
for id in station:
    print 'Station ID - %s' % id
```

get_statistics(*ts_from=0*, *ts_till=0*, *virus_limit=10*)

Makes request to a server and returns a dictionary with overall group statistics (including subgroups).

Parameters

- **ts_from** – timestamp of when statistics starts. If 0 - statistics starts at year 1970. Default 0.
- **ts_till** – timestamp of when staristics ends. If 0 - statistics ends today. Default 0.
- **virus_limit** – virus statistics limiter, to return no more than a given number of viruses. Default 10.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
stats = group.get_statistics()
```

An extract from dictionary returned:

```
{
    'viruses': [
        {
            'name': 'Trojan.Spambot',
            'count': 3,
            'objects_list': [
                {
                     'originator': 57,
                     'originator_text': 'SpIDer Guard G3 for Workstations',
                     'owner': 'Unknown',
                      ...
                },
                ...
            ],
        },
    'scans': {
        'files': 128,
        'cured': 28,
         ...
    },
    'infections': {
        'ignored': 0,
        'errors': 0,
        'deleted': 28,
         ...
    },
    'stations_total': 1,
    'stations_state': {
        'unactivated': 0,
        'deinstalled': 0,
        'online': 1,
        ...
    }
}
```

get_subgroups(*as_id=True*)

Returns a list of subgroups of a group, performing a server call.

Parameters **as_id** – boolean. If *True* method returns a list of subgroups IDs. If *False* - a list of `AVDeskGroup` objects.

Returns list

Raises `AVDeskError` on failure

Warning: Setting **as_id** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full group information. One should bear it in mind when querying groups with large amount of subgroups.

Example:

```
subgroups = resource_obj.get_subgroups()
for id in subgroups:
    print 'Subgroup ID - %s' % id
```

is_custom

This *property* is used to verify group is *custom* (artificial).

is_platform

This *property* is used to verify group is *platform* virtual group.

is_status

This *property* is used to verify group is *status* virtual group (includes stations Online, Offline, etc.).

is_system

This *property* is used to verify group is *system* group (not artificial).

is_tariff

This *property* is used to verify group is *tariff* group (given that tariffs are groups also).

is_transport

This *property* is used to verify group is *transport* virtual group (includes stations using TPC/IP, IPX, etc.).

is_virtual

This *property* is used to verify group is common *virtual* (dynamic group).

key

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

parent

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print 'Parent ID - %s' % parent
```

retrieve_info(*resource_id=None*)

Performs a server call to retrieve complete resource information by its ID and puts it into object's properties.

Parameters **resource_id** – specific ID of the resource. If *None*, ID is taken from the object itself.

Raises `AVDeskError` on failure

Example:

```
# resource_obj contains no additional info.
assert resource_obj.name is None

# After the following request
resource_obj.retrieve_info('some_resource_id')

# resource_obj contains additional info.
assert resource_obj.name is not None
```

save(*auto_retrieve=True*)

A convinience method that automatically creates new resource on server if it doesn't exists or updates it if it does.

Under the hood it switches between `create()` and `update()` methods.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *save* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

> **Warning:** Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won’t be used further after the operation.

Example:

```
saved = resource_obj.save()
```

send_message (*message_text*, *url=’‘*, *url_text=’‘*, *logo_abs_path=’‘*, *logo_text=’‘*, *logo_url=’‘*)

Sends a message to all group participants.

Parameters

- **message_text** – Message text which can contain {link} macros to place link given in link parameters to
- **url** – URL to be placed in {link} macros
- **url_text** – URL description text to be placed in {link} macros
- **logo_abs_path** – Absolute path to a logo .bmp file to show in message
- **logo_text** – Description text for a logo
- **logo_url** – URL to go to on a logo click

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
group.send_message('Hello from pyavdesk!')
```

set_av_component (*component_id*, *components_state=1*)

Sets information about av component for the group.

Parameters

- **component_id** – component identifier from AV_COMPONENT_IDS.
- **components_state** – component state identifier from AV_COMPONENT_STATES.

Raises `AVDeskError` on failure

set_parent (*id_or_obj*)

Sets parent resource for the current resource.

Parameters **id_or_obj** – Parent resource object or ID.

Raises `AVDeskError` on failure

Example:

```
child_resource_obj.set_parent(parent_resource_obj)
# Is equivalent to:
child_resource_obj.set_parent('parent_resource_id')
```

statistics

This *property* is used to get group statistics. See `get_statistics()`.

update(*auto_retrieve=True*)

Performs server call in attempt to update the resource with information from class properties.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *update* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
updated = resource_obj.update()
```

3.7 Tariffs - AVDeskTariff class

Tariff pyavdesk resource described by **AVDeskTariff** class exposes an interface to manipulate tariffs at server.

Note: It is advised to use server resource class **AVDeskServer** as a single entry point for every server resource manipulation instead of direct instantiation of **AVDeskTariff** class.

Basic class usage example:

```
# Setting up server connection parameters.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')

# Get basic tariff.
tariff_classic = av_server.get_tariff(pyavdesk.TARIFF_IDS['CLASSIC'])
print tariff_classic.name

# Create a new tariff.
new_tariff = av_server.new_tariff('My new tariff')
new_tariff.save()
```

class pyavdesk.pyavdesk.**AVDeskTariff**(*connector_resource*, *resource_id=None*, *predefined_handle=None*)

AV-Desk tariff group resource class is used to perform tariff groups manipulations.

Parameters

- **connector_resource** – should be `AVDeskServer` instance.
- **resource_id** – if set server call is performed to get information for resource with given ID.
- **predefined_handle** – if set, resource data could be fetched from a resource at given handle.

Raises `AVDeskError` on failure

add_emails(*emails_list*)

Adds given emails to a list of e-mails defined for the resource.

Note: This method does not send data to server, for this to be done one needs to save resource object.

Parameters **emails_list** – list of e-mails

Example:

```
resource_obj.add_emails(['person1@server.com', 'person2@server.com'])
```

add_station(*station_obj*)

Adds a given station to current tariff group making a server call. Creates current tariff group on server if it doesn't already exist.

Parameters **station_obj** – `AVDeskStation` instance

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
my_station = av_server.new_station('My station')
saved = tariff.add_station(my_station)
```

create(*auto_retrieve=True*)

Performs a server call to create resource with properties defined in object.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *create* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
created = tariff.create()
```

delete(*stations_move_to=None*)

Performs a server call in an attempt to delete tariff.

Note: Stations connected to this tariff (if any) should be moved to another tariff group (see `stations_move_to` parameter).

Warning: `stations_move_to` parameter set to None only works for tariff groups that do not contain any stations already.

Parameters **stations_move_to** – tariff ID to move stations to.

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
deleted = tariff.delete('move_to_tariff_id')
```

delete_emails(*emails_list*)

Removes given emails from a list of e-mails defined for the resource.

> **Note:** This method does not send data to server, for this to be done one needs to save resource object.

Parameters **emails_list** – list of e-mails

Example:

```
resource_obj.delete_emails(['person1@server.com', 'person2@server.com', 'person3@server.com'
```

get_administrators(*as_logins=True*)

Returns a list of administators bound to a group, performing a server call.

Parameters **as_logins** – boolean. If *True* method returns a list of logins. If *False* - a list of `AVDeskAdmin` objects.

Returns list

Raises `AVDeskError` on failure

> **Warning:** Setting **as_logins** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full administrator information. One should bear it in mind when querying groups with large amount of administrators.

Example:

```
administrators = resource_obj.get_administrators()
for login in administrators:
    print 'An administrator with login - %s' % login
```

get_av_components()

Performs a server call to retrieve a list of antivirus application components information for the resource.

Returns list of dictionaries with components data

Raises `AVDeskError` on failure

Example:

```
components = resource_obj.get_av_components()
```

An extract from returned list:

```
[
    {
        'status': 1,
        'code': 105,
        'parent_group_id': '2888b7ff-3625-465e-bcb8-957de17f6458',
        'code_text': 'Dr.Web Firewall',
        'status_text': 'Optional',
        ...
    },
    ...
]
```

get_av_rights()

Performs a server call to retrieve a list of rights defined for the resource.

Returns list of dictionaries with rigths data

Raises `AVDeskError` on failure

Example:

```
rights = resource_obj.get_av_rights()
```

An extract from returned list:

```
[
    {
        'status': 1,
        'code': 53,
        'parent_group_id': '2888b7ff-3625-465e-bcb8-957de17f6458',
        'code_text': 'Uninstall Dr.Web Agent',
        'status_text': 'Enabled',
        ...
    },
    ...
]
```

get_emails()

Returns a list of e-mails defined for the resource.

Returns list of e-mails

Example:

```
emails = resource_obj.get_emails()
```

get_key()

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

get_parent(*as_id=True*)

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print ’Parent ID - %s’ % parent
```

get_resource_id()

Helper method to get resource identifier, which can be passed to such resource manipulation methods as `get_info()` and `delete()`.

Returns resource identifier as string

get_stations(*as_id=True*)

Returns a list of stations in a group, performing a server call.

Parameters **as_id** – boolean. If *True* method returns a list of stations IDs. If *False* - a list of `AVDeskStation` objects.

Returns list

Raises `AVDeskError` on failure

> **Warning:** Setting **as_id** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full station information. One should bear it in mind when querying groups with large amount of stations.

Example:

```
stations = resource_obj.get_stations()
for id in station:
    print ’Station ID - %s’ % id
```

get_subgroups(*as_id=True*)

Returns a list of subgroups of a group, performing a server call.

Parameters **as_id** – boolean. If *True* method returns a list of subgroups IDs. If *False* - a list of `AVDeskGroup` objects.

Returns list

Raises `AVDeskError` on failure

> **Warning:** Setting **as_id** parameter to *False* may lead to considerable server load, since separate server call is performed to retrieve full group information. One should bear it in mind when querying groups with large amount of subgroups.

Example:

```
subgroups = resource_obj.get_subgroups()
for id in subgroups:
    print ’Subgroup ID - %s’ % id
```

grace_period

This *property* is should be an integer from 0 to 99, denoting number of free (no-pay) days included into the tariff.

key

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

parent

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print 'Parent ID - %s' % parent
```

retrieve_info(*resource_id=None*)

Performs a server call to retrieve complete resource information by its ID and puts it into object's properties.

Parameters **resource_id** – specific ID of the resource. If *None*, ID is taken from the object itself.

Raises `AVDeskError` on failure

Example:

```
# resource_obj contains no additional info.
assert resource_obj.name is None

# After the following request
resource_obj.retrieve_info('some_resource_id')

# resource_obj contains additional info.
assert resource_obj.name is not None
```

save(*auto_retrieve=True*)

A convinience method that automatically creates new resource on server if it doesn't exists or updates it if it does.

Under the hood it switches between `create()` and `update()` methods.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *save* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
saved = resource_obj.save()
```

set_av_component (*component_id*, *components_state=1*)
Sets information about av component for the group.

Parameters

- **component_id** – component identifier from AV_COMPONENT_IDS.
- **components_state** – component state identifier from AV_COMPONENT_STATES.

Raises `AVDeskError` on failure

set_parent (*id_or_obj*)
Sets parent resource for the current resource.

Parameters id_or_obj – Parent resource object or ID.

Raises `AVDeskError` on failure

Example:

```
child_resource_obj.set_parent(parent_resource_obj)
# Is equivalent to:
child_resource_obj.set_parent('parent_resource_id')
```

update (*auto_retrieve=True*)
Performs server call in attempt to update the resource with information from class properties.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters auto_retrieve – boolean to specify whether an additional call to server is required after the *update* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
updated = tariff.update()
```

3.8 Stations - AVDeskStation class

pyavdesk station resource described by **AVDeskStation** class exposes an interface to manipulate antivirus stations (PCs).

Note: It is advised to use server resource class **AVDeskServer** as a single entry point for every server resource manipulation instead of direct instantiation of **AVDeskStation** class.

Basic class usage example:

```
# Setting up server connection parameters.
av_server = pyavdesk.AVDeskServer('admin', 'password', 'http://192.168.1.70')

# Create a new station.
new_station = av_server.new_station()
new_station.save()

# Output generated station ID.
print new_station.id
```

class pyavdesk.pyavdesk.**AVDeskStation**(*connector_resource*, *resource_id=None*, *predefined_handle=None*)

AV-Desk station resource class is used to perform stations manipulations.

Parameters

- **connector_resource** – should be `AVDeskServer` instance.
- **resource_id** – if set server call is performed to get information for resource with given ID.
- **predefined_handle** – if set, resource data could be fetched from a resource at given handle.

Raises `AVDeskError` on failure

add_emails(*emails_list*)

Adds given emails to a list of e-mails defined for the resource.

Note: This method does not send data to server, for this to be done one needs to save resource object.

Parameters **emails_list** – list of e-mails

Example:

```
resource_obj.add_emails(['person1@server.com', 'person2@server.com'])
```

add_to_groups(*groups_list*)

Add groups to resource association by group IDs or objects.

Parameters **groups_list** – list of `AVDeskGroup` instances, or a list of strings representing groups IDs

Raises `AVDeskError` on failure

Note: Resource to group association is not sent to server until object's `save()` is not called.

Warning: Some AV-Desk server versions (namely 6.2) might handle only one group per administrator. In that case only the first group from the given list is linked with the administrator.

Example:

```
resource_obj.add_to_groups(['my_group_id', my_another_group_obj])
resource_obj.save()
```

create(*auto_retrieve=True*)

Performs a server call to create resource with properties defined in object.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *create* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

> **Warning:** Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
created = resource_obj.create()
```

`delete`(***kwargs*)

Performs server call in an attempt to delete the resource.

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
deleted = resource_obj.delete()
```

`delete_emails`(*emails_list*)

Removes given emails from a list of e-mails defined for the resource.

> **Note:** This method does not send data to server, for this to be done one needs to save resource object.

Parameters **emails_list** – list of e-mails

Example:

```
resource_obj.delete_emails(['person1@server.com', 'person2@server.com', 'person3@server.com'
```

`delete_from_groups`(*groups_list*)

Removes groups to resource association by group IDs or objects.

Parameters **groups_list** – list of `AVDeskGroup` instances, or a list of strings representing groups IDs

Raises `AVDeskError` on failure

> **Note:** Resource to group association is not sent to server until object's `save()` is not called.

Example:

```
resource_obj.delete_from_groups(['my_group_id', my_another_group_obj])
resource_obj.save()
```

`download_url`

This *property* is used to get station's download URL as string.

`expires_time`

This *property* is used to get and set station's *expires* datetime (as unix timestamp).

`get_av_bases`()

Returns a list of antivirus bases used by station.

Returns list of dictionaries with av bases data

Raises `AVDeskError` on failure

Example:

```
av_bases = station.get_av_bases()
```

An extract from returned list:

```
[
    {
        ’file_name’: ’drw50001.vdb’,
        ’created_ts’: 1320059155,
        ’version’: ’500’,
        ’viruses’: 24512
    },
    ...
]
```

`get_av_components``()`

Performs a server call to retrieve a list of antivirus application components information for the resource.

Returns list of dictionaries with components data

Raises `AVDeskError` on failure

Example:

```
components = resource_obj.get_av_components()
```

An extract from returned list:

```
[
    {
        ’status’: 1,
        ’code’: 105,
        ’parent_group_id’: ’2888b7ff-3625-465e-bcb8-957de17f6458’,
        ’code_text’: ’Dr.Web Firewall’,
        ’status_text’: ’Optional’,
        ...
    },
    ...
]
```

`get_av_components_installed``()`

Returns a list of antivirus application components installed at the station.

Returns list of dictionaries with components data

Raises `AVDeskError` on failure

Example:

```
installed = station.get_av_components_installed()
```

An extract from returned list:

```
[
    {
        ’path’: ’C:\Program Files\DrWeb AV-Desk’,
        ’code’: 103,
        ’code_text’: ’Dr.Web Microsoft Outlook plugin’,
        'server': 'tcp/94.251.81.210:2193',
        'installed_ts': 1320062616
    },
    ...
]
```

get_av_components_running()

Returns a list of antivirus application components running at the station.

Returns list of dictionaries with components data

Raises AVDeskError on failure

Example:

```
running = station.get_av_components_running()
```

An extract from returned list:

```
[
    {
        'code': 30,
        'code_text': 'Dr.Web AV-Desk Agent for Windows',
        'started_ts': 1320523510,
        'params': None,
        'user': None,
        'type': 8
    },
    ...
]
```

get_av_modules()

Returns a list of antivirus modules used by the station.

Returns list of dictionaries with modules data

Raises AVDeskError on failure

Example:

```
modules = station.get_av_modules()
```

An extract from returned list:

```
[
    {
        'hash': '78a61d78e110fbf27975df4a4dc70dfc',
        'name': 'Dr.Web(c) Scanner for Windows',
        'created_ts': 1320062617,
        'modified_ts': 1320062530,
        'file_name': 'DRWEB32W.EXE',
        'version': '6.00.11.7112',
        'file_size': 2299656
    },
    ...
]
```

get_av_packages()

Returns a list of antivirus agent download URLs for various platforms.

Returns list of dictionaries with av packages data

Raises AVDeskError on failure

Example:

```
av_packages = station.get_av_packages()
```

An extract from returned list:

```
[
    {
        'url': 'http://localhost/download/download.ds?id=some_id',
        'type': 1
    },
    ...
]
```

get_av_rights()

Performs a server call to retrieve a list of rights defined for the resource.

Returns list of dictionaries with rigths data

Raises AVDeskError on failure

Example:

```
rights = resource_obj.get_av_rights()
```

An extract from returned list:

```
[
    {
        'status': 1,
        'code': 53,
        'parent_group_id': '2888b7ff-3625-465e-bcb8-957de17f6458',
        'code_text': 'Uninstall Dr.Web Agent',
        'status_text': 'Enabled',
        ...
    },
    ...
]
```

get_block_time()

Returns a tuple with information about station blocking: begin and end unix timestamps.

Returns list of dictionaries with modules data

Raises AVDeskError on failure

Example:

```
start_at, finish_at = station.get_block_time()
```

get_emails()

Returns a list of e-mails defined for the resource.

Returns list of e-mails

Example:

```
emails = resource_obj.get_emails()
```

get_groups(*as_id=True*)

Performs a server call and returns a list of groups associated with the resource.

Parameters **as_id** – boolean. If *True* list of `AVDeskGroup` instances is returned, if *False* - list of strings representing groups IDs

Returns list of objects which type is defined by *as_id* parameter

Raises `AVDeskError` on failure

Note: Objects in the list returned have only basic information. To get full infomation use `AVDeskGroup.retrieve_info()`.

Example:

```
groups = resource_obj.get_groups()
for group in groups:
    print 'Group ID - %s' % group.id
```

get_history (*ts_from=0*, *ts_till=0*, *event_type_filter=None*)

Performs a server call to retrieve station history information.

Parameters

- **ts_from** – timestamp of when history starts. If 0 - history starts at year 1970. Default 0.
- **ts_till** – timestamp of when history ends. If 0 - history ends today. Default 0.
- **event_type_filter** – event type identifier (see `HISTORY_EVENT_TYPES`) to filter history upon.

Returns list of dictionaries with events data

Raises `AVDeskError` on failure

Example:

```
history = station.get_history()
```

An extract from returned list:

```
[
    {
        'created_ts': 4294967295,
        'action_ts': 4294967295,
        'action_start_ts': 0,
        'action_finish_ts': 0,
        'tariff': None,
        'tariff_name': None,
        'event_type_id': 5
    },
    ...
]
```

get_key ()

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

get_parent(*as_id=True*)

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print 'Parent ID - %s' % parent
```

get_place_data()

Performs a server call to retrieve station's place information as dictionary.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
place = station.get_place_data()
```

An extract from dictionary returned:

```
{
    'province': 'Academgorodok',
    'city': 'Novosibirsk',
    'street': 'Some street',
    'room': None,
    'organization': 'Doctor Web, Ltd.',
    'latitude': 0,
    'longitude': 0,
    'country': 643,
    'floor': None,
    'department': None
}
```

get_resource_id()

Helper method to get resource identifier, which can be passed to such resource manipulation methods as `get_info()` and `delete()`.

Returns resource identifier as string

get_statistics(*ts_from=0*, *ts_till=0*, *virus_limit=10*)

Makes request to a server and returns a dictionary with station statistics.

Parameters

- **ts_from** – timestamp of when statistics starts. If 0 - statistics starts at year 1970. Default 0.
- **ts_till** – timestamp of when staristics ends. If 0 - statistics ends today. Default 0.
- **virus_limit** – virus statistics limiter, to return no more than a given number of viruses. Default 10.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
stats = station.get_statistics()
```

An extract from dictionary returned:

```
{
    'viruses': [
        {
            'name': 'Trojan.Spambot',
            'count': 3,
            'objects_list': [
                {
                    'originator': 57,
                     'originator_text': 'SpIDer Guard G3 for Workstations',
                     'owner': 'Unknown',
                      ...
                },
                ...
            ],
        },
    'scans': {
        'files': 128,
        'cured': 28,
         ...
    },
    'infections': {
        'ignored': 0,
        'errors': 0,
        'deleted': 28,
         ...
    },
    'stations_total': 1,
    'stations_state': {
        'unactivated': 0,
        'deinstalled': 0,
        'online': 1,
        ...
    }
}
```

get_tariff(*as_id=True*)

Performs a server call to retrieve a tariff resource for the station.

Parameters **as_id** – boolean. If *True* `AVDeskTariff` object is returned, if *False* - tariff ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
tariff_id = resource_obj.get_tariff()
print 'Station tariff ID - %s' % tariff_id
```

grace_period

This *property* is used to get station's grace period (in days) inherited from tariff group.

key

Performs a server call to retrieve a dictionary with key information for the resource.

Returns dictionary with key information

Raises `AVDeskError` on failure

Example:

```
key = resource_obj.get_key()
```

An example of returned dictionary:

```
{
    'inherited_group_id': '20e27d73-d21d-b211-a788-85419c46f0e6',
    'key': '=?ASCII?B?OyBEcldlYjMyIHY0LjE2?=' # This is a key in Base64 format.
}
```

last_seen_addr

This *property* is used to get station's last seen network address as string.

last_seen_time

This *property* is used to get station's last seen unix timestamp.

os

This *property* is used to get station's OS as string.

parent

Performs a server call to retrieve a parent resource for the current resource.

Parameters **as_id** – boolean. If *True* resource object is returned, if *False* - resources' ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
parent = resource_obj.get_parent()
print 'Parent ID - %s' % parent
```

password

This *property* is used to get and station's password.

place_data

Performs a server call to retrieve station's place information as dictionary.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
place = station.get_place_data()
```

An extract from dictionary returned:

```
{
    'province': 'Academgorodok',
    'city': 'Novosibirsk',
    'street': 'Some street',
    'room': None,
    'organization': 'Doctor Web, Ltd.',
    'latitude': 0,
    ’longitude’: 0,
    ’country’: 643,
    ’floor’: None,
    ’department’: None
}
```

retrieve_info(*resource_id=None*)

Performs a server call to retrieve complete resource information by its ID and puts it into object’s properties.

Parameters **resource_id** – specific ID of the resource. If *None*, ID is taken from the object itself.

Raises `AVDeskError` on failure

Example:

```
# resource_obj contains no additional info.
assert resource_obj.name is None

# After the following request
resource_obj.retrieve_info(’some_resource_id’)

# resource_obj contains additional info.
assert resource_obj.name is not None
```

save(*auto_retrieve=True*)

A convinience method that automatically creates new resource on server if it doesn’t exists or updates it if it does.

Under the hood it switches between `create()` and `update()` methods.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *save* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

> **Warning:** Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won’t be used further after the operation.

Example:

```
saved = resource_obj.save()
```

send_message(*message_text*, *url=’‘*, *url_text=’‘*, *logo_abs_path=’‘*, *logo_text=’‘*, *logo_url=’‘*)

Sends a message to all group participants.

Parameters

- **message_text** – Message text which can contain {link} macros to place link given in link parameters to
- **url** – URL to be placed in {link} macros
- **url_text** – URL description text to be placed in {link} macros
- **logo_abs_path** – Absolute path to a logo .bmp file to show in message
- **logo_text** – Description text for a logo

- **logo_url** – URL to go to on a logo click

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
group.send_message('Hello from pyavdesk!')
```

set_block_time(*begin_at*, *finish_at*)
Sets blocking information for the station.

Parameters

- **begin_at** – Block begin unix timestamp
- **finish_at** – Block end unix timestamp

Returns *True* on success

Raises `AVDeskError` on failure

Example:

```
station.set_block_time(start_at, finish_at)
```

set_parent(*id_or_obj*)
Sets parent resource for the current resource.

Parameters **id_or_obj** – Parent resource object or ID.

Raises `AVDeskError` on failure

Example:

```
child_resource_obj.set_parent(parent_resource_obj)
# Is equivalent to:
child_resource_obj.set_parent('parent_resource_id')
```

set_place_data(*country=None*, *latitude=None*, *longitude=None*, *province=None*, *city=None*, *street=None*, *organization=None*, *department=None*, *floor=None*, *room=None*)
Sets information about station's place.

Parameters

- **country** – ISO numeric country code integer - (see http://en.wikipedia.org/wiki/ISO_3166-1_numeric)
- **latitude** – Latitude as integer
- **longitude** – Latitude as integer
- **province** – Province name string
- **city** – City name string
- **street** – Street name string
- **organization** – Organization name string
- **department** – Department name string
- **floor** – Floor string
- **room** – Room string

Example:

```
# Selectively set country and organization name for the station.
station.set_place_data(country=643, organization='Doctor Web, Ltd.')
```

set_tariff(*id_or_obj*)

Sets tariff for the current station.

Parameters **id_or_obj** – Tariff resource object or ID.

Raises `AVDeskError` on failure

Example:

```
station.set_tariff(tariff_resource_obj)
# Is equivalent to:
station.set_tariff('tariff_resource_id')
```

state

This *property* is used to get station's current state as string (e.g. *online*).

statistics

Makes request to a server and returns a dictionary with station statistics.

Parameters

- **ts_from** – timestamp of when statistics starts. If 0 - statistics starts at year 1970. Default 0.
- **ts_till** – timestamp of when staristics ends. If 0 - statistics ends today. Default 0.
- **virus_limit** – virus statistics limiter, to return no more than a given number of viruses. Default 10.

Returns dictionary

Raises `AVDeskError` on failure

Example:

```
stats = station.get_statistics()
```

An extract from dictionary returned:

```
{
    'viruses': [
        {
            'name': 'Trojan.Spambot',
            'count': 3,
            'objects_list': [
                {
                    'originator': 57,
                     'originator_text': 'SpIDer Guard G3 for Workstations',
                     'owner': 'Unknown',
                      ...
                },
                ...
            ],
        },
    'scans': {
        'files': 128,
        'cured': 28,
         ...
    },
    'infections': {
        'ignored': 0,
        'errors': 0,
        'deleted': 28,
         ...
    },
    'stations_total': 1,
    'stations_state': {
        'unactivated': 0,
        'deinstalled': 0,
        'online': 1,
        ...
    }
}
```

tariff

Performs a server call to retrieve a tariff resource for the station.

Parameters **as_id** – boolean. If *True* `AVDeskTariff` object is returned, if *False* - tariff ID

Returns object or ID. See `as_id` parameter.

Raises `AVDeskError` on failure

Example:

```
tariff_id = resource_obj.get_tariff()
print 'Station tariff ID - %s' % tariff_id
```

update(*auto_retrieve=True*)

Performs server call in attempt to update the resource with information from class properties.

Note: There is a convinience method `save()` to handle both `create` and `update` operations.

Parameters **auto_retrieve** – boolean to specify whether an additional call to server is required after the *update* operation to retrieve full resource data. Default: True.

Returns *True* on success

Raises `AVDeskError` on failure

Warning: Setting `auto_retrieve` to False may increase operation performance, but also may leave the resource object data in a not up-to-date state. It is advised that `auto_retrieve` set to False is only used when the resource object won't be used further after the operation.

Example:

```
updated = resource_obj.update()
```

CHAPTER 4

Indices and tables

- *genindex*
- *search*