SpaceEZ Documentation

Release v1.0.0

Juniper Networks Inc.

July 13, 2015

Contents

CHAPTER 1

Class Index

jnpr.space.rest.Space(url[, user, passwd, ...])	Encapsulates a Junos Space cluster and provides access to all RESTfu
jnpr.space.resource.Resource(type_name, ...)	Represents a **resource** that is exposed by Junos Space REST API.
jnpr.space.collection.Collection(parent, ...)	Represents a **collection** that is exposed by Junos Space REST API.
jnpr.space.method.Method(parent, name, mobj)	Represents a **method** that is exposed by Junos Space REST API.
jnpr.space.service.Service(rest_end_point, ...)	Represents a **service** that is exposed by Junos Space REST API.
jnpr.space.application.Application(...)	Represents an **application** that is hosted on Junos Space platform.
jnpr.space.async.TaskMonitor(rest_end_point, ...)	Encapsulates the logic required to monitor the progress of tasks using

CHAPTER 2

Module Index

jnpr.space.factory	A module with functions using which new resource objects can be created.
jnpr.space.xmlutil	A module with utility functions for XML handling.

CHAPTER 3

Rest

This module defines the Space class.

exception jnpr.space.rest.**RestException**(*message*, *response*)

An exception that is raised when a REST API invocation returns a response indicating an error. The exception contains the complete response inside.

Attributes: response: The response object (requests.Response) which resulted in this exception.

__weakref__

list of weak references to the object (if defined)

class jnpr.space.rest.**Space**(*url*, *user=None*, *passwd=None*, *cert=None*, *use_session=False*, *required_node=None*, *profile_file=None*)

Encapsulates a Junos Space cluster and provides access to all RESTful web-service APIs provided by Space. An instance of this class is also referred to as a *'rest end point'* in this documentation.

Note: In most cases, you don't directly invoke methods of this class. The typical usage pattern is to create an instance of this class and then access contained services and collections and invoke methods on them.

For example, the snippet below creates a Space instance and then performs a GET on the devices collection contained by the device-management web service:

```
>>> s = rest.Space(url='https://1.1.1.1',
                   user='super',
                   passwd='password')
>>> devs = s.device_management.devices.get()
```

Note: Instances of this class are thread-safe. So you can have multiple threads invoking APIs on the same Junos Space cluster using one instance of this class.

__getattr__(*attr*)

This method is overridden in the class so that applications & services contained by this instance can be accessed as *normal* Python attributes of this object.

Parameters **attr** (*str*) – Name of the app or web-service being accessed.

Returns jnpr.space.service.Service or jnpr.space.application.Application being accessed.

Raises AttributeError if there is no application or service with the given name.

__getitem__(*attr*)

This method is overridden so that contained elements can be accessed using their 'xml names' - e.g. user['first-name']. The implementation just calls __getattr__ internally.

See doc for __getattr__ for more details.

__init__(*url*, *user=None*, *passwd=None*, *cert=None*, *use_session=False*, *required_node=None*, *profile_file=None*)
Creates an instance of this class to represent a Junos Space cluster.

Parameters

- **url** (*str*) – URL of the Junos Space cluster using its VIP address. E.g. https://<VIP>
- **user** (*str*) – A valid userid for invoking APIs on this Space cluster. Can be omitted if `cert` is provided for X.509 certificate based authentication to Junos Space.
- **passwd** (*str*) – Password for the userid.
- **cert** (*tuple*) – X.509 certificate details for authentication. This is to be used only to access Junos Space that is configured to perform X.509 certificate based authentication. Otherwise, this parameter defaults to `None`. If used, it MUST be a tuple that contains (1) the full pathname of the X.509 certificate PEM file; and (2) the full pathname of the X.509 certificate key file. It is recommended that the key file is unencrypted - otherwise the SSL layer will prompt you to enter the passphrase used for encrypting key file and this prompt will appear for each SSL connection.
- **use_session** (*bool*) – Whether to use a session based login or not. It is `False` by default.
- **required_node** (*str*) – This parameter is used only if `use_session` is set to True. This is used to specify the name of the Junos Space node (e.g. space-000c2980f778) in the cluster on which the session should be established. This parameter is `None` by default.
- **profile_file** (*str*) – Full pathname of a file where response times for each API call is to be recorded. This parameter is `None` by default.

Returns An instance of this class encapsulating the Junos Space cluster whose **url** was given as a parameter. It can be used to access all APIs provided by Space.

delete(*delete_url*)
Performs an HTTP DELETE on the given url. Acts as a wrapper over requests.delete() function.

Parameters **url** (*str*) – URL for performing DELETE

Returns

The response object (requests.Response) returned by the POST request.

get(*url*, *headers={}*)
Performs an HTTP GET on the given url. Acts as a wrapper over requests.get() function.

Parameters

- **url** (*str*) – URL for performing GET
- **headers** (*dict*) – A dict with the headers that need to be sent with the GET request. Defaults to `{}`.

Returns

The response object (requests.Response) returned by the GET request.

head(*url*, *headers={}*)
Performs an HTTP HEAD on the given url. Acts as a wrapper over requests.head() function.

Parameters

- **url** (*str*) – URL for performing HEAD

- **headers** (*dict*) – A dict with the headers that need to be sent with the HEAD request. Defaults to `{}`.

Returns

The response object (requests.Response) returned by the HEAD request.

login(*required_node=None*)

Logs into Space and creates a session (connection) that is maintained. All API calls will use this session and will use the JSESSIONID, JSESSIONIDSSO cookies - they will not be individually authenticated.

Parameters **required_node** (*str*) – This is used to specify the name of the Junos Space node in the cluster on which the session should be established. It is `None` by default.

logout()

Logs out the current session being used.

post(*url*, *headers*, *body*)

Performs an HTTP POST on the given url. Acts as a wrapper over requests.post() function.

Parameters

- **url** (*str*) – URL for performing POST
- **headers** (*dict*) – A dict with the headers that need to be sent with the POST request. This is a mandatory parameter.
- **body** (*str*) – A string that forms the body of the POST request. This is a mandatory parameter.

Returns

The response object (requests.Response) returned by the POST request.

put(*put_url*, *headers*, *body*)

Performs an HTTP PUT on the given url. Acts as a wrapper over requests.put() function.

Parameters

- **url** (*str*) – URL for performing PUT
- **headers** (*dict*) – A dict with the headers that need to be sent with the PUT request. This is a mandatory parameter.
- **body** (*str*) – A string that forms the body of the PUT request. This is a mandatory parameter.

Returns

The response object (requests.Response) returned by the PUT request.

CHAPTER 4

Resource

This module defines the Resource class.

class jnpr.space.resource.**MetaResource**(*app_name*, *service_name*, *key*, *values*)
Encapsulates the meta data for a resource type.

create_collection(*resrc*, *name*)
Creates a collection object.

Parameters

- **resrc** (jnpr.space.resource.Resource) – Parent resource.
- **name** (*str*) – Name of the collection.

Returns A jnpr.space.collection.Collection object.

create_method(*resrc*, *name*)
Creates a method object.

Parameters

- **resrc** (jnpr.space.resource.Resource) – Parent resource.
- **name** (*str*) – Name of the method.

Returns A jnpr.space.method.Method object.

get_media_type(*version*)
Returns media-type modelled in the yaml file.

class jnpr.space.resource.**Resource**(*type_name*, *rest_end_point*, *xml_data=None*, *attributes=None*, *parent=None*)
Represents a **resource** that is exposed by Junos Space REST API. Some examples of resources are:

•A device (/api/space/device-management/devices/{id})

•Configuration of a device (/api/space/device-management/devices/{id}/configurations/raw)

•A user (/api/space/user-management/users/{id})

•A tag (/api/space/tag-management/tags/{id})

delete()
Deletes this resource on Space by sending a DELETE request with the url of this resource.

Returns None

Raises jnpr.space.rest.RestException if the DELETE method results in an error response. The exception's response attribute will have the full response from Space.

form_xml()
Forms an XML representation of the state of this resource based on its attribute values.

Returns An `lxml.etree.Element` object representing the state of this resource.

get(*attr=None*, *accept=None*)
This is an overloaded method that does two things: If the `attr` parameter is passed, it returns the corresponding XML attribute from the top level XML data element contained by this resource. If the `attr` parameter is not passed, it performs an HTTP GET for this resource and get its current state.

Parameters

- **attr** (*str*) – The name of the XML attribute in the top-level element of the XML state of the resource. Defaults to `None`.
- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the GET request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.

Returns

- Value of the named XML attribute. OR
- The current state of this resource fetched from Space and represented as a Python object. You can access the fields of the resource's state directly as attributes of this object.

Raises `jnpr.space.rest.RestException` if the GET method results in an error response. The exception's `response` attribute will have the full response from Space.

get_href()
Gets the href for this resource. If `href` is available as an attr inside _xml_data, it is returned. Otherwise, this method checks if _xml_data has a valid `uri` attr and if so, returns that. Otherwise, it checks if the _parent reference is set and if yes, it will concat the parent's href with the `id` attr from _xml_data. If all this fail, then it concatentates `service_url` and `collection_name` from its meta object and appends the `id` attr from _xml_data.

Returns The href of this resource.

get_meta_object()
Returns the meta object which holds meta data for this resource.

Returns `jnpr.space.resource.MetaResource`

post(*accept=None*, *content_type=None*, *request_body=None*, *task_monitor=None*, *schedule=None*, **args*, ***kwargs*)
Some resources support the POST method. For example, the configuration of a device supports the POST method which can be used to fetch selected portions of the configuration based on xpath expressions. On such resources, this method can be used to send the POST request.

Parameters

- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **content_type** (*str*) – This can be used to supply a media-type that must be used as the Content-Type header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **request_body** (*str*) – This can be used to supply a string that must be used as the request body in the request. This defaults to `None` and in this case SpaceEZ will create the request body using the modeled template, replacing variables with kwargs.

- **task_monitor** (jnpr.space.async.TaskMonitor) – A TaskMonitor object that can be used to monitor the progress of the POST request, in case of asynchronous invocations. You need to check Junos Space API documentation to see if the POST invocation on this resource has asynchronous semantics and supply the task_monitor parameter only if it is asynchronous. Otherwise, this will default to `None` and the method will behave with synchronous semantics.
- **schedule** (*str*) – A string specifying a cron expression for scheduling the execution of the request on the Space server side. This is applicable only if the POST invocation on this resource has asynchronous semantics and you want to schedule the execution. Otherwise, this will default to `None`.
- **kwargs** (*A variable list of name=value arguments.*) – Keyword args of the form name=value which will be used to substitute variables in a pre-defined template (if applicable) to form the request body.

Returns A Python object constructed from the response body that Space returned for the POST method invocation. In the case of asynchronous invocation, this will represent a Task object and will contain the unique id of the Task executing the POST request in Space.

Raises `jnpr.space.rest.RestException` if the POST method results in an error response. The exception's `response` attribute will have the full response from Space.

put(*new_val_obj=None*, *request_body=None*, *accept=None*, *content_type=None*)

Modifies the state of this resource on Space by sending a PUT request with the new state. The attributes of *new_val_obj* are formatted to form the XML request body for the PUT request. If the parameter *new_val_obj* is `None`, then the argument *request_body*, if present, is used as the request body. If this is also `None`, then the attributes of this object itself are formatted to form the XML request body. Once the PUT request successfully completes, it re-initializes the state of this Resource object based on the XML response from Space.

Parameters

- **new_val_obj** (jnpr.space.resource.Resource) – A Resource object with the newly desired state for the resource. This defaults to `None`.
- **request_body** (*str*) – This can be used to supply a string that must be used as the request body in the request. This defaults to `None` and in this case SpaceEZ will create the request body using the supplied `new_val_obj` argument or from the current state of this object.
- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **content_type** (*str*) – This can be used to supply a media-type that must be used as the Content-Type header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.

Returns `None`

Raises `jnpr.space.rest.RestException` if the PUT method results in an error response. The exception's `response` attribute will have the full response from Space.

state()

Prints the XML string into stdout.

xml_data()

Returns a formatted string that represents the state of this resource.

xml_string()
Returns a string that contains formatted XML representing the state of this resource.

jnpr.space.resource.**get_meta_object**(*full_name*, *values*)
Looks up the meta object for a resource based on its fully qualified type name of the form <service-name>.<type_name> or <app-name>.<service-name>.<type_name>

Parameters **full_name** (*str*) – Fully qualified type name of the resource.

Returns A jnpr.space.resource.MetaResource object.

CHAPTER 5

Collection

This module defines the Collection class.

class jnpr.space.collection.**Collection**(*parent*, *name*, *meta_object*)

Represents a **collection** that is exposed by Junos Space REST API. Some examples of collections are:

- The devices collection (/api/space/device-management/devices)
- The users collection (/api/space/user-management/users)
- The tags collection (/api/space/tag-management/tags)

get(*accept=None*, *filter_=None*, *domain_id=None*, *paging=None*, *sortby=None*)

Gets the contained resources of this collection from Space.

Parameters

- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the GET request. This defaults to None and in this case SpaceEZ will use the media-type modeled in the description file.
- **filter** (*str or dict*) – A filter expression to apply on the collection. This can be given as a dict with name:value pairs to filter with. For example, {'name':'user1'}. Or this can be given as a string which forms a valid filter expression for this collection as per Junos Space API documentation. This parameter defaults to None.
- **paging** (*dict*) – A paging expression to apply on the collection. This must be given as a dict with entries giving values for start and limit paging parameters. For example, {'start':10, 'limit':100}. This parameter defaults to None.
- **sortby** (*list of str*) – A list of field names to sort the results by. This parameter defaults to None.

Returns A list of jnpr.space.resource.Resource objects.

Raises jnpr.space.rest.RestException if the GET method results in an error response. The exception's response attribute will have the full response from Space.

get_href()

Gets the href for this collection. If the meta object has a url set, it is returned. Otherwise, it concatenates the href of the parent object and the name of this collection to form the href and returns it.

Returns The href of this collection.

post(*new_obj=None*, *accept=None*, *content_type=None*, *request_body=None*, *xml_name=None*, *task_monitor=None*)

Sends a POST request to the Space server to create a new Resource in this collection.

Parameters

- **new_obj** (A single `jnpr.space.resource.Resource` instance or a list of them.) – The new Resource that needs to be created as a member of this collection. This can be omitted in which case the caller must supply a request body for the POST request as the `request_body` argument.
- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **content_type** (*str*) – This can be used to supply a media-type that must be used as the Content-Type header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **request_body** (*str*) – This can be used to supply a string that must be used as the request body in the request. This defaults to `None` and in this case the caller must supply the `new_obj` argument from which the request body will be formed.
- **xml_name** (*str*) – Can be used to override the name of the top-level XML element in the generated request body. This is useful in some cases such as creating a quick config template. This parameter defaults to `None`.
- **task_monitor** (jnpr.space.async.TaskMonitor) – A TaskMonitor object that can be used to monitor the progress of the POST request, in case of asynchronous invocations. You need to check Junos Space API documentation to see if the POST invocation on this resource has asynchronous semantics and supply the task_monitor parameter only if it is asynchronous. Otherwise, this will default to `None` and the method will behave with synchronous semantics.

Returns If the new_obj parameter is a list, then the same list is returned. Otherwise, this method creates a new Resource object based on the state of the newly created resource, as extracted from the POST response body. In the case of asynchronous invocation, this will represent a Task object and will contain the unique id of the Task executing the POST request in Space.

Raises `jnpr.space.rest.RestException` if the POST method results in an error response. The exception's `response` attribute will have the full response from Space.

state()
Performs a GET on this collection to fetch the current state and prints it as XML.

class `jnpr.space.collection.`**MetaCollection**(*app_name*, *service_name*, *key*, *values*)
Encapsulates the meta data for a collection type.

create_method(*service*, *name*)
Creates a method object corresponding to the given service and name.

Parameters

- **service** (jnpr.space.service.Service) – Parent service.
- **name** (*str*) – Name of the method.

Returns A `jnpr.space.method.Method` object.

get_media_type(*version*)
Returns the media-type defined inside the meta object.

class `jnpr.space.collection.`**ResourceList**(*resource_list*)
Encapsulates a list of Resource objects and provides methods to print the state of all of them.

state()
Prints the XML string into stdout.

xml_data()
Returns a formatted string that represents the state of all resources in this list.

xml_string()
Returns a string that contains formatted XML representing the state of all resources in this list.

jnpr.space.collection.**get_meta_object**(*app_name*, *service_name*, *coll_name*, *values*)
Looks up the meta object for a collection based on its fully qualified type name of the form <service-name>.<coll_name> or <app-name>.<service-name>.<coll-name>.

Parameters

- **app_name** (*str*) – Name of the application.
- **service_name** (*str*) – Name of the service.
- **coll_name** (*str*) – Name of the collection.

Returns A jnpr.space.collection.MetaCollection object.

CHAPTER 6

Method

This module defines the Method class.

class jnpr.space.method.**MetaMethod**(*key*, *values*)
Encapsulates the meta data for a method.

get_media_type(*version*)
Returns media type modeled in yaml file.

get_request_type(*version*)
Returns request media type modeled in yaml file.

get_response_type(*version*)
Returns response media type modeled in yaml file.

class jnpr.space.method.**Method**(*parent*, *name*, *mobj*)
Represents a **method** that is exposed by Junos Space REST API. Some examples of methods are:

- exec-rpc on a device (/api/space/device-management/devices/{id}/exec-rpc)
- change-password on a user account (/api/space/user-management/users/{id}/change-password)

get(*accept=None*)
Performs a GET corresponding to the Method object.

Parameters **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the GET request. This defaults to None and in this case SpaceEZ will use the media-type modeled in the description file.

Returns A Python object constructed from the response body that Space returned for the GET request.

Raises jnpr.space.rest.RestException if the GET results in an error response. The exception's response attribute will have the full response from Space.

get_href()
Gets the href for this method. If the meta object has its name set as '-', this returns the href of the parent. Otherwise, it concatenates the href of the parent object and the name of this method to form the href and returns it.

Returns The href of this method.

post(*accept=None*, *content_type=None*, *request_body=None*, *task_monitor=None*, *schedule=None*, **args*, ***kwargs*)
This sends a POST request corresponding to this Method object.

Parameters

- **accept** (*str*) – This can be used to supply a media-type that must be used as the Accept header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **content_type** (*str*) – This can be used to supply a media-type that must be used as the Content-Type header in the request. This defaults to `None` and in this case SpaceEZ will use the media-type modeled in the description file.
- **request_body** (*str*) – This can be used to supply a string that must be used as the request body in the request. This defaults to `None` and in this case SpaceEZ will create the request body using the modeled template, replacing variables with kwargs.
- **task_monitor** (jnpr.space.async.TaskMonitor) – A TaskMonitor object that can be used to monitor the progress of the POST request, in case of asynchronous invocations. You need to check Junos Space API documentation to see if the POST invocation on this resource has asynchronous semantics and supply the task_monitor parameter only if it is asynchronous. Otherwise, this will default to `None` and the method will behave with synchronous semantics.
- **schedule** (*str*) – A string specifying a cron expression for scheduling the execution of the request on the Space server side. This is applicable only if the POST invocation on this resource has asynchronous semantics and you want to schedule the execution. Otherwise, this will default to `None`.
- **kwargs** (*A variable list of name=value arguments.*) – Keyword args of the form name=value which will be used to substitute variables in a pre-defined template to form the request body. The template name is specified inside the meta data for this Method object.

Returns A Python object constructed from the response body that Space returned for the POST method invocation. In the case of asynchronous invocation, this will represent a Task object and will contain the unique id of the Task executing the POST request in Space.

Raises `jnpr.space.rest.RestException` if the POST method results in an error response. The exception's `response` attribute will have the full response from Space.

`jnpr.space.method.`**get_meta_object**(*app_name*, *service_name*, *method_name*, *values*)

Looks up the meta object for a method based on its fully qualified type name of the form `<service-name>.<method_name>` or `<app-name>.<service-name>.<method-name>`.

Parameters

- **app_name** (*str*) – Name of the application.
- **service_name** (*str*) – Name of the service.
- **method_name** (*str*) – Name of the service.

Returns A `jnpr.space.method.MetaMethod` object.

CHAPTER 7

Service

This module defines the Service class.

class jnpr.space.service.**MetaService**(*name*, *values*, *application=None*)
Encapsulates the meta data for a service.

create_collection(*service*, *name*)
Creates a collection object corresponding to the given service and name.

Parameters

- **service** (*str*) – Name of the parent service.
- **name** (*str*) – Name of the collection.

Returns A jnpr.space.collection.Collection object.

create_method(*service*, *name*)
Creates a method object corresponding to the given service and name.

Parameters

- **service** (jnpr.space.service.Service) – Parent service.
- **name** (*str*) – Name of the method.

Returns A jnpr.space.method.Method object.

get_application_name()
Returns the name of the containing application if there is one.

get_meta_resource(*name*)
Returns the MetaResource object with the given name.

Parameters **name** (*str*) – Name of the resource.

Returns A jnpr.space.resource.MetaResource object.

class jnpr.space.service.**Service**(*rest_end_point*, *name*, *values*, *application=None*)
Represents a **service** that is exposed by Junos Space REST API. Some examples of services are:

•Device Management (/api/space/device-management)

•User Management (/api/space/user-management)

get_href()
Returns the href of this service.

get_meta_resource(*resource_type*)
Returns the MetaResource object with the given name.

Parameters **name** (*str*) – Name of the resource.

Returns A `jnpr.space.resource.MetaResource` object.

CHAPTER 8

Application

This module defines the Application class.

class jnpr.space.application.**Application**(*rest_end_point*, *name*, *values*)
Represents an **application** that is hosted on Junos Space platform. Some examples of applications are:

- Service Now (/api/juniper/servicenow)
- Service Insight (/api/juniper/serviceinsight)

get_href()
Returns the href of this application.

class jnpr.space.application.**MetaApplication**(*name*, *values*)
Encapsulates the meta data for an application.

CHAPTER 9

Async

This module defines the TaskMonitor class.

class jnpr.space.async.**TaskMonitor**(*rest_end_point*, *qname*, *wait_time=10*, *max_consecutive_attempts=10*)

Encapsulates the logic required to monitor the progress of tasks using a hornet-q. An instance of this class acts as a wrapper over a hornet-q. When you instantiate this class, it internally creates a hornet-q with the given `qname`. You can then use this `TaskMonitor` instance when invoking asynchronous APIs using the SpaceEZ library. Each asynchronous API invocation will create a new task (a.k.a Job) in Junos Space. You can wait for a single task to complete using the `wait_for_task` method which returns the `ProgressUpdate` message for the task with completion state and status of the task. If you use the same `TaskMonitor` object to create multiple tasks, you can wait for the completion of all of them using the `wait_for_tasks` method which returns a list of `ProgressUpdate` messages - one for each task.

The snippet below shows an example where a TaskMonitor is used in creating a discover-devices task and waiting for its completion:

```
>>> s = rest.Space(url='https://1.1.1.1',
                   user='super',
                   passwd='password')
>>> devs = s.device_management.devices.get()
>>> tm = async.TaskMonitor(s, 'test_DD_q')
>>> result = s.device_management.discover_devices.post(
        task_monitor=tm,
        hostName='test-host-name',
        manageDiscoveredSystemsFlag=True,
        userName='regress', password='MaRtInI')
>>> pu = tm.wait_for_task(result.id)
>>> print pu.state, pu.status
```

Note: If you use multi-threaded programming, make sure that you **do not** share a `TaskMonitor` instance across multiple threads. You should use separate `TaskMonitor` objects for separate threads.

__init__(*rest_end_point*, *qname*, *wait_time=10*, *max_consecutive_attempts=10*)

Creates an instance of this class to encapsulate a hornet-q.

Parameters

- **rest_end_point** – An instance of `rest.Space` class which encapsulates a Junos Space system on which tasks need to be monitored.
- **qname** (*str*) – The name of the underlying hornet-q.
- **wait_time** (*int*) – Number of seconds to sleep between trying to pull progress-update

messages from the hornet-q. This is also used as the value of the `accept-wait` header when pulling messages from the hornet-q. The default value of this argument is 10.

- **`max_consecutive_attempts`** (*int*) – The maximum number of consecutive attempts that will be made to pull progress-update messages from the hornet-q. This defaults to 10. We will consider the task as hanging if we are not able to pull a progress-update message from the hornet-q after this many attempts.

Returns An instance of this class encapsulating a hornet-q with the given name (`qname`).

`__module__` = 'jnpr.space.async'

`delete`()
Cleanup by deleting the hornet-q encapsulated by this object.

`get_final_progress_update`(*pu_message*)
Gets the final progress-update message for a job, based on the href inside the `pu_message` argument supplied. The `pu_message` arg is the last progress-update message obtained from a hornet-q and it does not contain the full result of the job. The progress-update message fetched and returned by this method will contain the full result inside the `data` field.

`get_queue_url`()
Returns the full URL for the encapsulated hornet-q

`pull_message`()
Pull the next message from the hornet-q. It specifies an `accept-wait` header with the value that was given as the `wait_time` argument in the constructor so that the pull waits for this many seconds on the server side waiting for a message.

Returns If a message is pulled, it is parsed into a Python object and returned. If no message was pulled, returns `None`.

`wait_for_task`(*task_id*)
Waits for the given task to complete by periodically pulling progress-update messages from the hornet-q and checking if the message indicates that the task is Done. The time between consecutive pulls is the value given as the `wait_time` argument in the constructor. The maximum number of consecutive pulls is determined by the `max_consecutive_attempts` argument in the constructor. If we're unable to pull a progress-update message from hornet-q after this many consecutive attempts, we will consider the task as hanging and this method will raise an Exception.

Note: You should use this method only when you have created one task using this TaskMonitor object. If you created multiple tasks, you must use the wait_for_tasks() method to wait for their completion.

Returns The final progress-update message for the task is fetched and returned inside a Python object. The `data` attribute of this object will contain the full result from the task.

`wait_for_tasks`(*task_id_list*)
Waits for all the tasks in the given list to complete by periodically pulling progress-update messages from the hornet-q and checking if the message indicates that any of the given tasks is Done. The time between consecutive pulls is the value given as the `wait_time` argument in the constructor.

The maximum number of consecutive pulls is determined by the `max_consecutive_attempts` argument in the constructor. If we're unable to pull a progress-update message from hornet-q after this many consecutive attempts, we will consider the tasks as hanging and this method will raise an Exception.

Returns A list of progress-update messages indicating the completion state and status of all the given tasks. Each entry in the list is a Python object representing the final progress-update message for a task. The `data` attribute of this object will contain the full result from the task.

CHAPTER 10

Factory

A module with functions using which new resource objects can be created. Resources are instances of jnpr.space.resource.Resource.

jnpr.space.factory.**fetch_resource**(*rest_end_point*, *href*)

This is the method you should use to create a Resource instance if you have the href for it. It gets the current state of the resource with the given *href*. Once the current state is fetched from the Space server, this method creates a new instance of `jnpr.space.resource.Resource` with this state and returns it.

Parameters

- **rest_end_point** (jnpr.space.rest.Space) – A *Space* object encapsulating the Junos Space cluster which contains this resource.
- **href** (*str*) – The href of the resource that needs to be fetched.

Returns A new instance of jnpr.space.resource.Resource

Raises `jnpr.space.rest.RestException` if the GET request results in an error response. The exception's `response` attribute will have the full response from Space.

jnpr.space.factory.**make_resource**(*type_name*, *rest_end_point*, *xml_data=None*, *attributes=None*, *parent=None*)

Creates a new instance of jnpr.space.resource.Resource based on the given parameters. This method creates it in-memory locally and *not* on the Space server. The post() method must be invoked on the Resource object to get it created on the Space server.

Parameters

- **type_name** (*str*) – Fully qualified type name for the Resource to be created. It is of the format `<service_name>.<resource_type>`. Some examples are:
  - `device_management.device`
  - `user_management.user`
- **rest_end_point** (jnpr.space.rest.Space) – A *Space* object encapsulating the Junos Space cluster which is to contain this resource.
- **xml_data** (*lxml.etree.Element*) – The state of the resource as an XML object. This defaults to `None`.
- **attributes** (*dict*) – The state of the resource as a dict where the keys are attribute names and values are attribute values. This defaults to `None`.
- **parent** (jnpr.space.collection.Collection) – The parent object of this resource. This defaults to `None`.

Returns A new instance of jnpr.space.resource.Resource

CHAPTER 11

XML Util

A module with utility functions for XML handling.

jnpr.space.xmlutil.**cleanup**(*src*)
Some responses from Space contains escaped form for XML special characters. E.g. '<' for '<', etc. This method removes these escaped notations.

Parameters **src** (*str*) – Source string

Returns String with escaped notations replaced.

jnpr.space.xmlutil.**get_text_from_response**(*response*)
Returns text from `Response` object that will be str (unicode) in both python 2 and 3.

jnpr.space.xmlutil.**get_xml_obj_from_response**(*response*)
Returns an XML object (`lxml.Element`) from the text inside the `Response` object.

jnpr.space.xmlutil.**make_xml_name**(*attr_name*)
Convert an attribute name to its XML equivalent by replacing all '_' with '-'. CamelCase names are retained as such.

Parameters **attr_name** (*str*) – Name of the attribute

Returns Attribute name in XML format.

jnpr.space.xmlutil.**remove_default_namespace**(*src*)
Remove default xmlns from the given string.

Parameters **src** (*str*) – Source string

Returns String with xmlns definitions removed.

jnpr.space.xmlutil.**remove_junos_group**(*src*)
Remove XML attribute junos:group from the given string.

Parameters **src** (*str*) – Source string

Returns String with junos:group occurrences removed.

jnpr.space.xmlutil.**unmake_xml_name**(*attr_name*)
Convert an XML attribute name to its pythonic equivalent by replacing all '-' with '_'. CamelCase names are retained as such.

Parameters **attr_name** (*str*) – Name of the attribute

Returns Attribute name in pythonic format.

jnpr.space.xmlutil.**xml2obj**(*src*)
Uses lxml.objectify to parse the given XML string and returns a Python object.

Parameters **src** (*str*) – Source XML string

Returns An instance of `lxml.objectify.ObjectifiedElement`

CHAPTER 12

Indices and tables

- genindex
- modindex
- search

j

Symbols

A

C

D

F

G

H

J

L

M

P

R

S

T

U

W

X