SharePlum Documentation

Release 0.4.1

Jason Rollins

Apr 22, 2020

Contents

SharePlum is an easier way to work with SharePoint services. It handles all of the messy parts of dealing with SharePoint and allows you to write clean and Pythonic code.

Example:

```
from shareplum import Site
from requests_ntlm import HttpNtlmAuth

cred = HttpNtlmAuth('Username', 'Password')
site = Site('https://mysharepoint.server.com/sites/MySite', auth=cred)
sp_list = site.List('list name')

list_data = sp_list.GetListItems()
```

CHAPTER 1

Installing

Installation is as easy as:

```
$ pip install shareplum
```

or install from source.

```
$ git clone git://github.com/jasonrollins/shareplum
$ cd shareplum
$ python setup.py install
```

1.1 Dependencies

SharePlum requires the following packages:

- lxml
- requests
- requests-ntlm
- requests-toolbelt

CHAPTER 2

Tutorial

2.1 On Premise Authentication

Getting started is easy. Just create some credentials you will use to connect to SharePoint with HttpNtlmAuth and pass the url and credentials to the Site object.

```
from shareplum import Site
from requests_ntlm import HttpNtlmAuth

cred = HttpNtlmAuth('Username', 'Password')
site = Site('https://mysharepoint.server.com/sites/MySite', auth=cred)
```

2.2 Office 365 Authentication

For Office 365 Sharepoint is just as easy. The Office365 class grabs a login token from Microsoft's login servers then It logins the Sharepoint site and uses the cookie for Authentication. Make sure to put just the root url for the site in Office365 and add Https:// at start.

```
from shareplum import Site
from shareplum import Office365

authcookie = Office365('https://abc.sharepoint.com', username='username@abc.com',␣
↪password='password').GetCookies()
site = Site('https://abc.sharepoint.com/sites/MySharePointSite/',␣
↪authcookie=authcookie)
```

2.3 Access REST API

You can access aditional features by utilizing the SharePoint REST API in SharePlum. To access this API you must specify a SharePoint version higher than 2013 when creating your Site object.

```
from shareplum import Site
from shareplum import Office365
from shareplum.site import Version

authcookie = Office365('https://abc.sharepoint.com', username='username@abc.com',␣
↪password='password').GetCookies()
site = Site('https://abc.sharepoint.com/sites/MySharePointSite/', version=Version.
↪v2016, authcookie=authcookie)
```

The REST API gives you access to the Folder class.

```
folder = site.Folder('Shared Documents/This Folder')
folder.upload_file('Hello', 'new.txt')
folder.get_file('new.txt')
folder.check_out('new.txt')
folder.check_in('new.txt', "My check-in comment")
folder.delete_file('new.txt')
```

2.4 SharePoint Versions

The available version options are given below:

```
Version.v2007 (default)
Version.v2010
Version.v2013
Version.v2016
Version.v2019
Version.v365
```

There are currently only 2 actual versions implemented: 2007 and 365. v2010 is an alias for v2007 and v2013, v2016, and v2019 are aliases for v365.

2.5 Add A List

You can easily create a new list for your site.

```
site.AddList('My New List', description='Great List!', templateID='Custom List')
```

2.6 Upload Data

Upload content to your list with the UpdateListItems method.

```
new_list = site.List('My New List')
my_data = data=[{'Title': 'First Row!'},
                {'Title': 'Another One!'}]
new_list.UpdateListItems(data=my_data, kind='New')
```

2.7 Download Data

Retrieve the data from a SharePoint list using GetListItems.

```
sp_data = new_list.GetListItems()
```

Retrieve the data from your list by specifiying a SharePoint View,

```
sp_data = new_list.GetListItems('All Items')
```

or specifying the fields you want.

```
sp_data = new_list.GetListItems(fields=['ID', 'Title'])
```

SharePlum will automatically convert the name of the column that is displayed when you view your list in a web browser to the internal SharePoint name so you don't have to worry about how SharePoint stores the data.

2.8 Update List Data

You can update data in a SharePoint List easily as well. You just need the ID number of the row you are updating.

```
update_data = [{'ID': '1', 'Title': 'My Changed Title'},
               {'ID': '2', 'Title': 'Another Change'}]
new_list.UpdateListItems(data=update_data, kind='Update')
```

CHAPTER 3

Queries

With SharePlum you can retrieve list items by providing a View Name or by providing a list of Column Names (fields) and a query. You don't have to provide a query. Given a list of Column Names, SharePlum will return all of the data for those columns. When you need to filter down this information, you can provide a query. A query has three major elements: Where, OrderBy, and GroupBy.

3.1 Where

The Where Element is probably the most commonly used.

```
fields = ['Title', 'My Other Column']
query = {'Where': [('Eq', 'My Other Column', 'Nice Value')]}
sp_data = sp_list.GetListItems(fields=fields, query=query)
```

You don't pass a value if you are using IsNull.

```
query = {'Where': [('IsNull', 'My Other Column')]}
sp_data = sp_list.GetListItems(fields=fields, query=query)
```

You can use AND or OR for multiple conditions

```
query = {'Where': ['And', ('Eq', 'Title', 'Good Title'),
                          ('Eq', 'My Other Column', 'Nice Value')]}

query = {'Where': ['Or', ('Eq', 'Title', 'Good Title'),
                         ('Eq', 'My Other Column', 'Nice Value')]}
```

or use them both

```
query = {'Where': ['Or',  ('Eq', 'My Other Column', 'Great Title'),
                   'And', ('Eq', 'My Other Column', 'Good Title'),
                          ('Eq', 'My Other Column', 'Nice Value')]}
```

3.1.1 Where Options:

- Eq: Equals
- Neq: Not Equal To
- Geq: Greater Than or Equal To
- Gt: Greater Than
- Leq: Less Than or Equal To
- Lt: Less Than
- IsNull: Value Is Null
- IsNotNull: Value is not Null
- BeginsWith: Begins With Text
- Contains: Contains Text

3.2 OrderBy

With OrderBy you can provide a list of Columns that will be used to sort your data. Ascending order is the default. If that's all you need, you can just provide the Column Names:

```
query = {'OrderBy': ['Title']}
```

If you want Descending order, you'll have to specify it.

```
query = {'OrderBy': [('Title', 'DESCENDING')]}
```

3.3 GroupBy

GroupBy can be used to group your data by Columns:

```
query = {'GroupBy': ['Title']}
```

CHAPTER 4

Files

SharePlum can work with files and folders in SharePoint version 2013 and higher using the REST API. To access this API you need to specify your SharePoint version when creating your Site instance:

```
site = Site('https://abc.sharepoint.com/sites/MySharePointSite/', version=Version.
↪v2016, authcookie=authcookie)
```

4.1 Folders

When you create an instance of a folder, you specifiy the folder location. This folder will be created if it does not exist.

```
folder = site.Folder('Shared Documents/This Folder')
```

4.2 Files

You can upload a file to the folder with upload_file()

```
folder.upload_file('Hello', 'new.txt')
```

or for a binary file

```
with open(fileName, mode='rb') as file:
        fileContent = file.read()
folder.upload_file(fileContent, "filename.bin")
```

Download a file

```
folder.get_file('source.txt')
```

Check out a file

```
folder.check_out('new.txt')
```

Check in a file

```
folder.check_in('new.txt', "My check-in comment")
```

Delete a file

```
folder.delete_file('new.txt')
```

CHAPTER 5

Advanced

5.1 SSL Version

Some Linux distributions using OpenSSL 1.0f or older can not use the TLS1.2 protocal as outlined here. You can change the SSL/TLS protocol version by passing in the ssl_version parameter for Site like so:

```
site = Site(SITE, auth=auth, verify_ssl=True, ssl_version='TLSv1')
```

CHAPTER 6

Classes and Methods

6.1 Site

The main object of the SharePlum library is Site.

class Site(*url* [*version=Version.v2007, auth=None, authcookie=None, verify_ssl=True, ssl_version='TLSv1', huge_tree=False, timeout=None*])

Main Site object used to interact with your SharePoint site.

6.1.1 Methods

AddList(*listName*, *description*, *templateID*)

Adds a list to your site with the provided name, description, and template.

Valid Templates include:

- Announcements
- Contacts
- Custom List in Datasheet View
- DataSources
- Discussion Board
- Document Library
- Events
- Form Library
- Issues
- Links
- Picture Library
- Survey

- Tasks

DeleteList (*listName*)
Delete a list on your site with the provided List Name.

GetListCollection ()
Returns information about the lists for the Site.

GetUsers ([*rowlimit=0*])
Returns information on the userbase of the current Site.

List (*listName*, *exclude_hidden_fields=False*)
Returns a List object for the list with ‘listName’ on the current Site.

Sometimes internal fields can take the same DisplayName as visible fields, effectively hiding them from SharePlum. When ‘exclude_hidden_fields’ is True, these internal fields won’t be loaded.

6.2 List

The List object contains methods for interacting with SharePoint Lists. Created with Site.List().

6.2.1 Methods

GetListItems ([*view_name=None*, *fields=None*, *query=None*, *row_limit=0*])

- viewname - A valid View Name for the current List.
- fields - Instead of a View we can pass the individual columns we want.
- query - A filter we can apply.
- rowlimit - Limit the number of rows returned

GetList ()
This is already run when the List object is initialized. You can access the returned data under self.schema

GetView (*viewname*)
Information about the provided View Name for the current list.

GetViewCollection ()
This is already run when the List object is initialized. You can access the returned data under self.views

UpdateList ()
Does nothing. TODO.

UpdateListItems (*data*, *kind*)
Add or edit data on the current List.

- data - Python Dictionary eg.:

```
data = {'Movie': 'Elf', 'Length': '1h 37min'}
```

- kind - ‘New’, ‘Update’, or ‘Delete’

When using kind=’Delete’ the data parameter becomes a list of IDs. eg.:

```
data = ['46', '201', '403', '456']
```

GetAttachmentCollection (*_id*)
Get a list of attachements for the row with the provided ID.

6.3 Folder

The Folder object is only usable with the REST API by specifiing site(version=Version.v2013) or greater.

`class Filder`(*folder_name*)

6.3.1 Attributes

`contextinfo`

`items`

`files`

6.3.2 Methods

`get_file`(*file_name*)

`upload_file`(*content*, *file_name*)

`check_out`(*file_name*)

`check_in`(*file_name*, *comment*)

`delete_folder`(*relative_url*)

`delete_file`(*file_name*)

6.4 soap

Helper class to build our SOAP requests. You shouldn't have to use this directly.

CHAPTER 7

Changelog

7.1 0.4.0

REST API support

7.2 0.3.0

Code cleanup and bug fixes.

7.3 0.1.1

Added support for Python 2.7

7.4 0.1.0

Initial Release.

Index