**Connect to Jenkins Utility using Python**

Implemented the following

Python Jenkins is a python wrapper for the Jenkins REST API which aims to provide a more conventionally pythonic way of controlling a Jenkins server. It provides a higher-level API containing a number of convenience functions.

We like to use python-jenkins to automate our Jenkins servers. Here are some of the things we used it for:

========Jenkins api list===========

* userInfo
* jobCount
* createJob
* getJobConfig
* buildJob
* disableJob
* copyJob
* enableJob
* reconfigJob
* deleteJob
* buildInfo
* createView
* getJobs
* ViewConfigs
* getViews
* deleteView
* pluginsInfo

**1.Get user Info** (userInfo)

Get information about the user account that authenticated to Jenkins. This is a simple way to verify that your credentials are correct.

**2. Job Count(jobCount)**

Lists the number of jobs available on the server.

**3. Create Job(createJob)**

Creates new job.

**4. Get Job Configuration(getJobConfig)**

Gets the configuration of the job.

**5. Build job (buildJob)**

Trigger build job.

This method returns a queue item number that you can pass to Jenkins.get\_queue\_item(). Note that this queue number is only valid for about five minutes after the job completes, so you should get/poll the queue information as soon as possible to determine the job’s URL.

**6. Disable Job(disableJob)**

Disables a Job.

**7. Copy Job (copyJob (from\_name, to\_name))**

Copy a Jenkins job.

Will raise an exception whenever the source and destination folder for this jobs won’t be the same.

**8. Enable Job(enableJob)**

Enables a Job again.

**9. Reconfigure Job(reconfigJob)**

Change configuration of existing Jenkins job.

**10. Delete Job(deleteJob)**

Deletes the Job specified.

**11. Build Info(buildInfo)**

Gives the Info of the build like Last Completed, last build number and so on.

**12. Create View(createView)**

Creates a new View

**13. Get Jobs(getJobs)**

Gets all the jobs

**14. View Configurations(viewConfig)**

Lists the configuration of the view

**15.Get views get\_views()**

Get list of views running.

Each view is a dictionary with ‘name’ and ‘url’ keys.

**16. Delete Views(deleteViews)**

Deletes the view

**17.Get Plugin information (pluginsInfo)**

Get all installed plugins information on this Master.

This method retrieves information about each plugin that is installed on master returning the raw plugin data in a JSON format.

Deprecated since version 0.4.9: Use get plugins() instead.

**To Implement the Jenkins using REST API in Python follow the below steps**

1.Download and install Jenkins on your local machine or you can also have access to your organizations server.

2. Create new account (if locally installed on your machine) or get account credentials from your organization.

3. Create an Api-Token or password for authentication.

4.Thats it happy hacking with APIs.

**On your Laptop / Computer perform following steps**

1. Install Python 3.4 and above version on System

2.Install a Python code editor (I’m using Pycharm Community Edition )

3.Install required packages and extensions on editor and system

4.Open py-jenkins.py (filename.py) file in your editor

5.Check Auth and URL and enter it accordingly as per you

6.If required packages and extensions are not installed

Run command (windows 10)

py -m pip install python-jenkins

7.Then run command py filename.py (py-jenkins.py) (windows 10)

Note : if you are working on different operating systems there are different commands other than windows 10 try “python “ instead of “py”

This is for server version of Jenkins