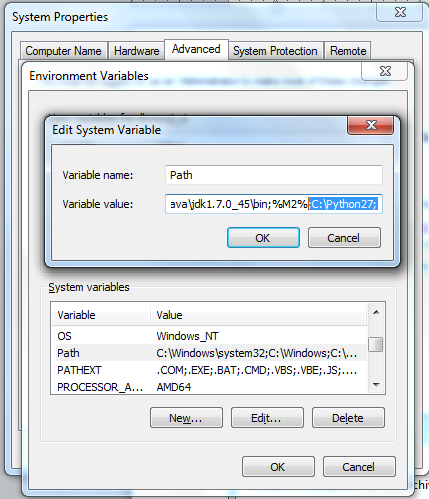
**Installing Python and other required packages**

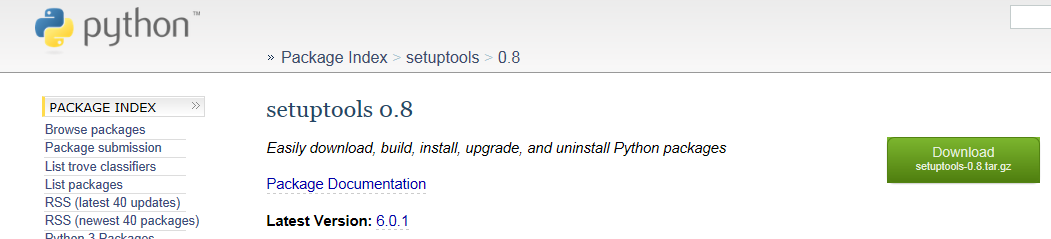
**Step 1: Install the Python 2.7.8**

1. Download and install the python2.7.8 from below link <https://www.python.org/download/releases/2.7.8/>
2. After installation we will find a folder named **Python27** in C drive
3. Update the value of path variable of system variables in Environment variable window. PFB the screen shot:



**Step 2: Install the Requests Package and lxml Package in Python**

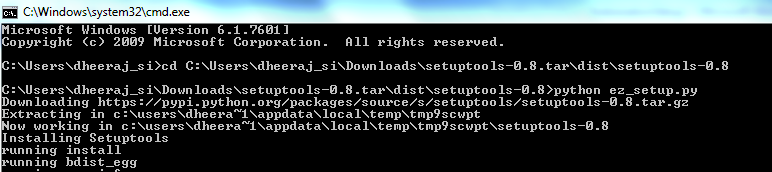
1. Download setuptools-0.8.tar.gz file from the below link : <https://pypi.python.org/pypi/setuptools/0.8> or <https://pypi.python.org/pypi/setuptools/0.8#downloads>

PFB the screen shot: 

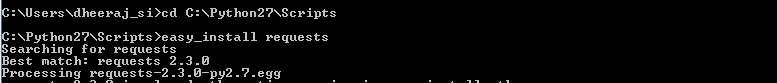
Note: If below commands don’t work install <https://confluence.gtie.dell.com/x/wQysDg>

Download Microsoft Visual C++ compiler package for Python 2.7

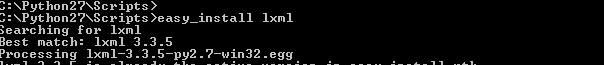
1. Download and extract the **setuptools-0.8.tar.gz** in a folder location say **setuptools-0.8**
2. Open the **cmd** and go to the extracted location and type **python ez\_setup.py** PFB the screen shot:



1. After installation of this, a folder named **Scripts** will be created in **C:\Python27**
2. Now Go to **C:\Python27\Scripts**, we will Find **'easy\_install'** exe file.
3. Now go to **Install requests package** in python Go to CMD and browse to Location C:\Python27\Scripts and type **'easy\_install requests'** PFB the screen shot:



1. Now go to **Install lxml package** in python Go to CMD and browse to Location C:\Python27\Scripts and type **'easy\_install lxml** PFB the screen shot:

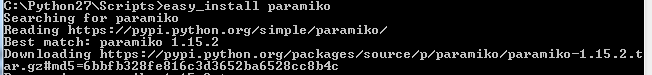


1. Now go to **Install xmlutils package** in python Go to CMD and browse to Location C:\Python27\Scripts and type **'easy\_install xmlutils** PFB the screen shot:



1. Now go to **Install paramiko package** in python Go to CMD and browse to Location C:\Python27\Scripts and type **'easy\_install paramiko** PFB the screen shot:

If u get error run “pip install paramiko” .. If pip is not installed download get-pip.py from <https://bootstrap.pypa.io> ; then run the command python python get-pip.py



1. Now go to **Install PyCrypto package** in python. Before installing **PyCrypto** please check the python version and bit from the command” **python –v** ” . For example in my case I have install python 2.7 64 bit so I have install

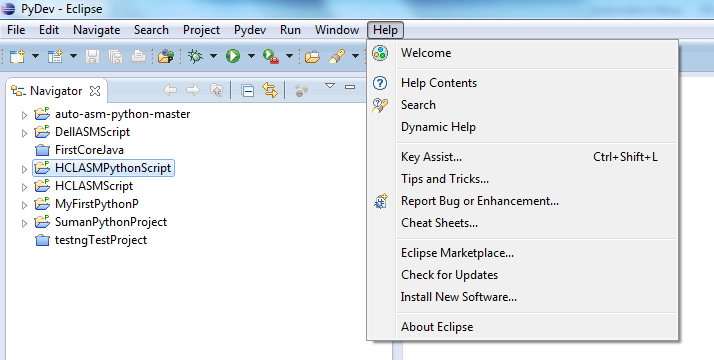


From the link <http://www.voidspace.org.uk/python/modules.shtml>

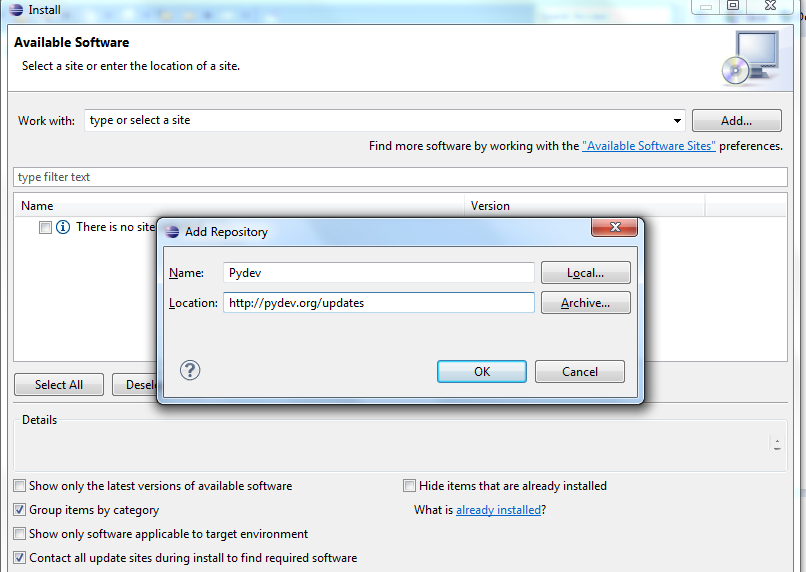
**Installation and Configuration of Pydev in Eclipse**

**Step 1: Installation of Pydev in Eclipse:**

1. Open the eclipse and then go to Help → Install New Software

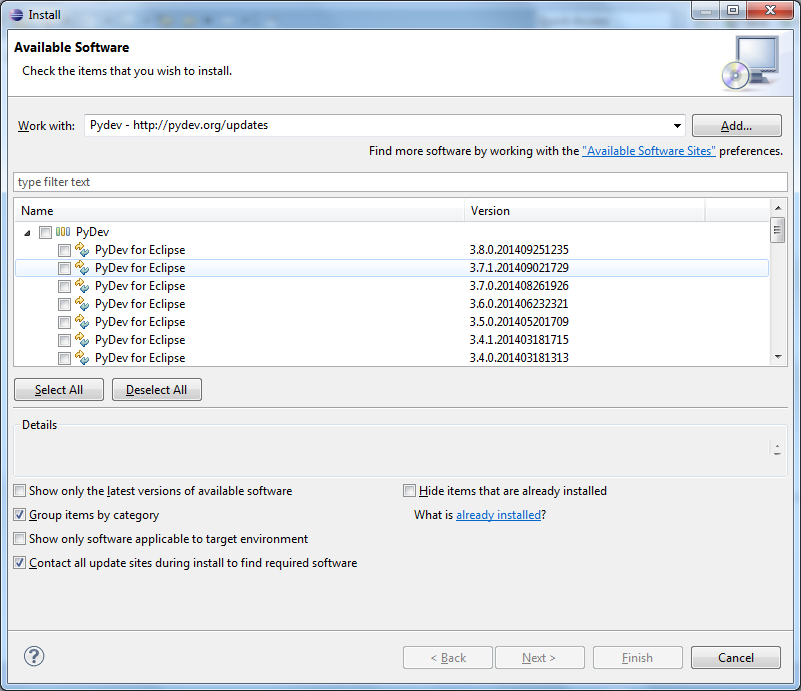


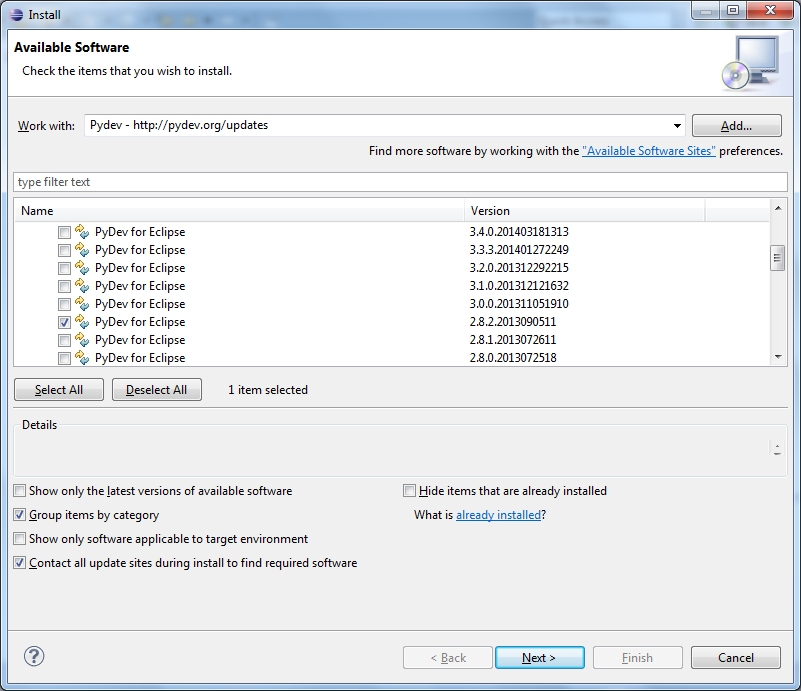
1. Install window will open and click on the add button. In the Name field write the **pydev** and in the Location field write the **http://pydev.org/updates** PFB the screen shot:



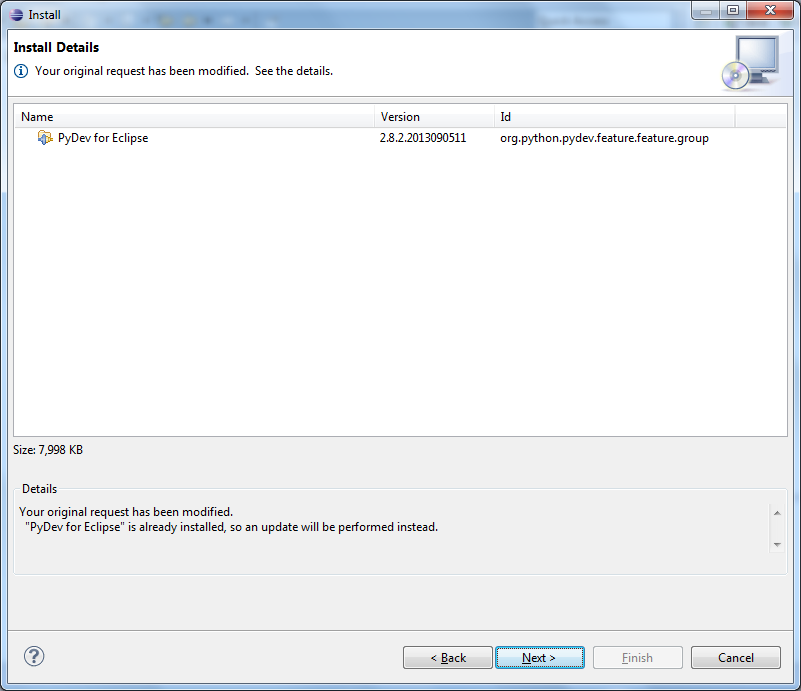
1. Once we click on the ok the new screen will look like Below Screen shot:

**Note: Please follow the below screens**

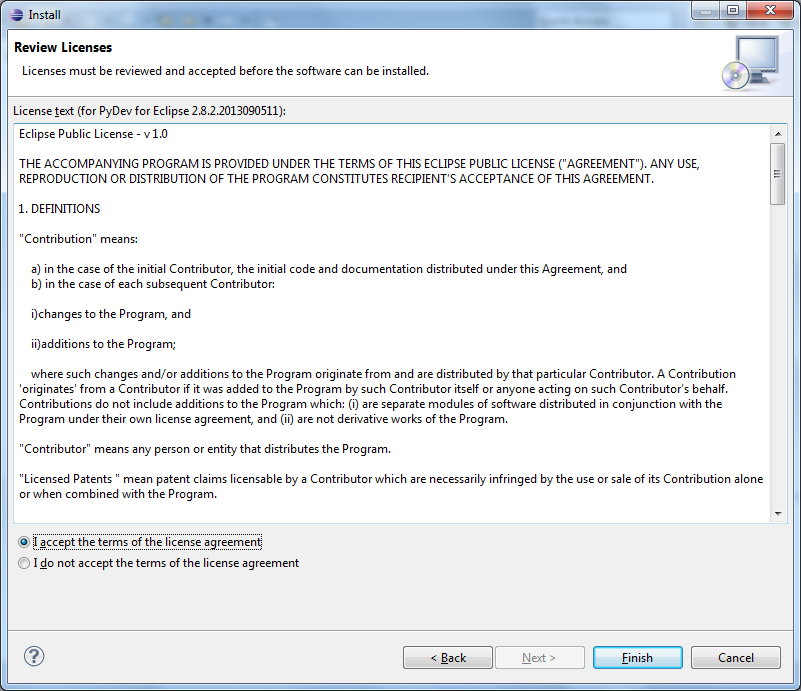




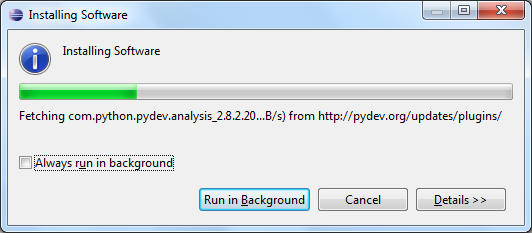
**Please only select the version 2.8.2 and click on next**



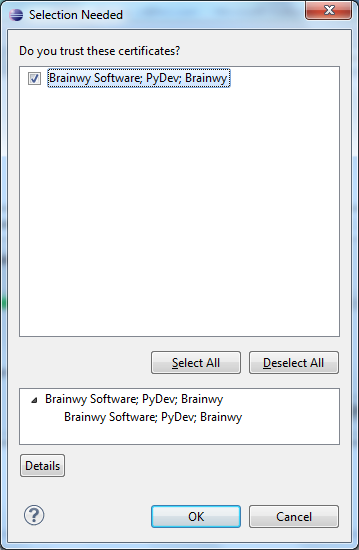
**Click on next**



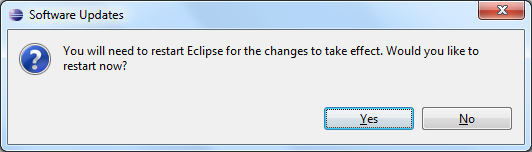
**Accept the license agreement and click on Finish.**



**It will takes some time.**



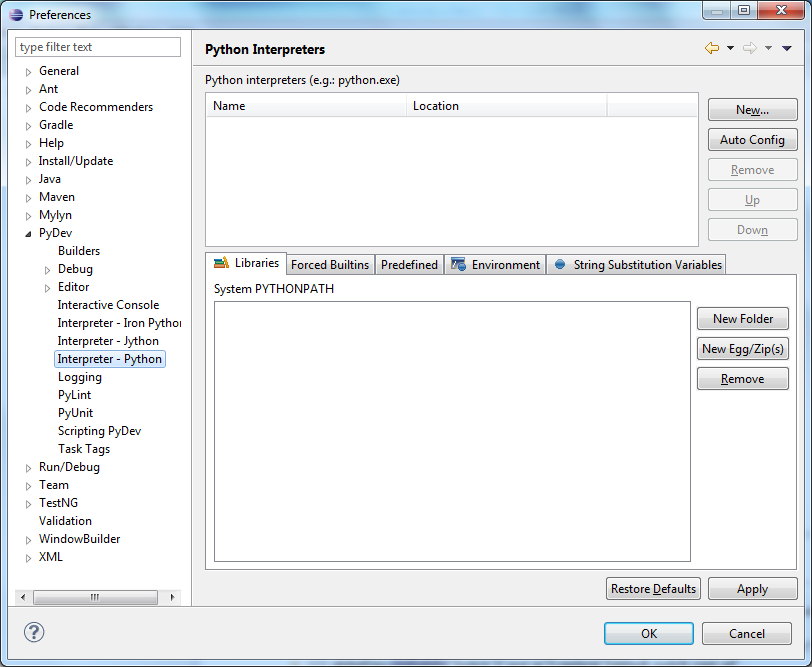
**Select the checkbox and click on ok**



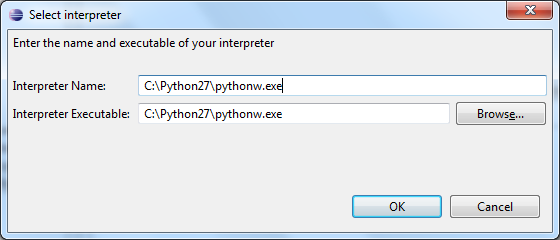
**Note: Once we get this screen means our pydev has been installed and now we need to configure the pydev.**

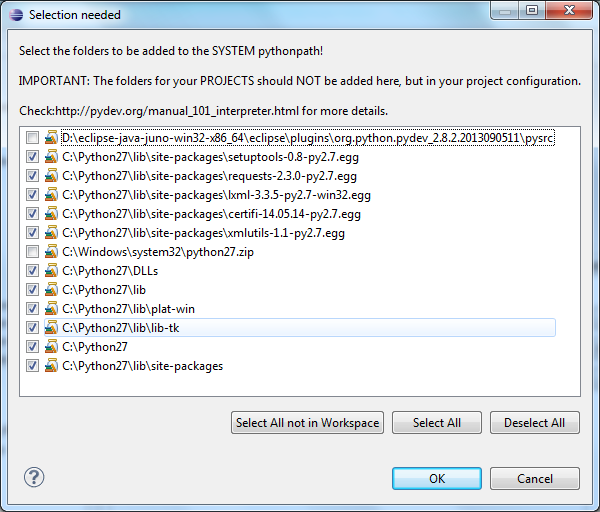
**Step 2: Configuration of Pydev in Eclipse:**

1. Open in the Window → Preference → Pydev → Interpreter Python menu.

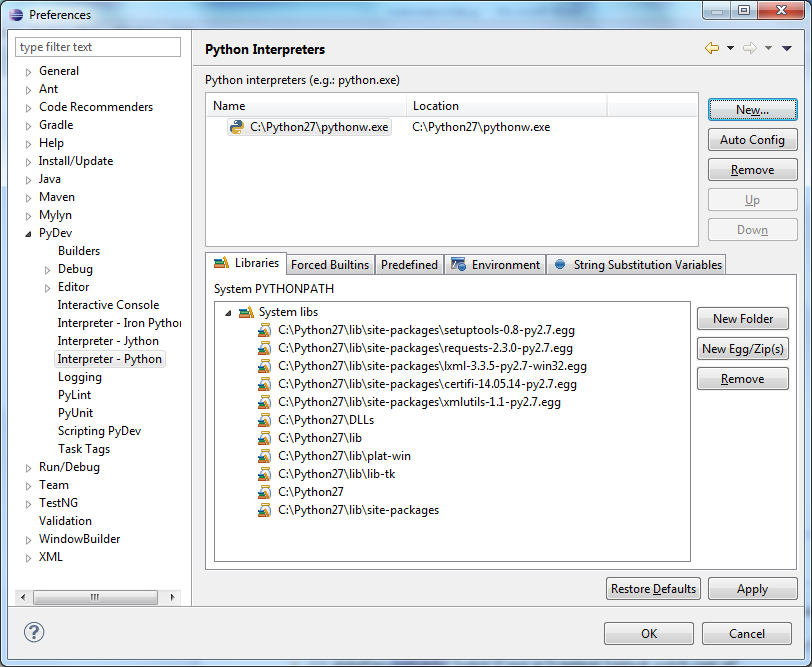


1. Press the *New* button and enter the path to python.exe in your Python installation directory.





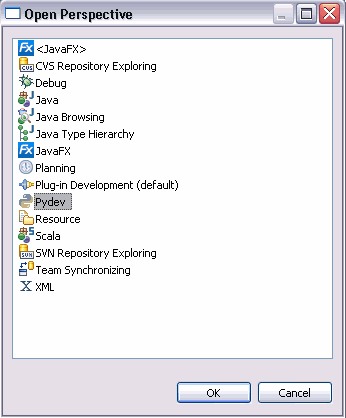
1. Select only checkboxes as shown in above and click ok.
2. The result should look like the following.



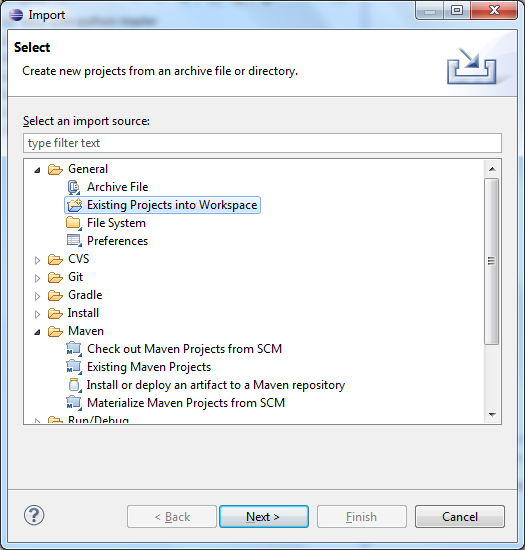
**Note:** Now you have successfully configured the pydev in eclipse

**Importing the Python Project in eclipse**

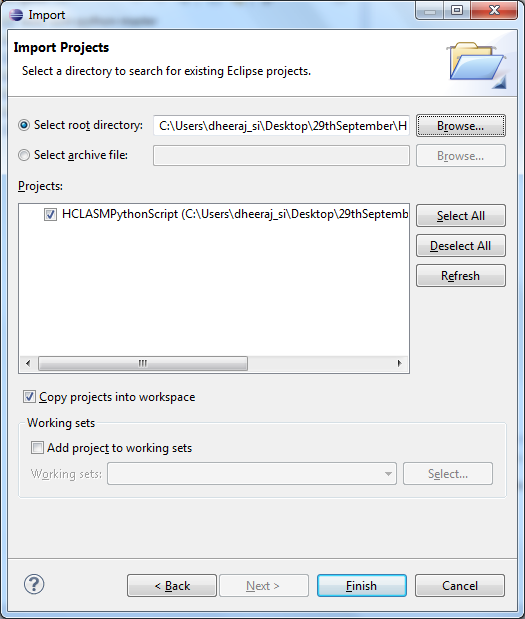
1. Open the eclipse and select an workspace
2. Select Window->Open Perspective ->Other. Select the PyDev perspective



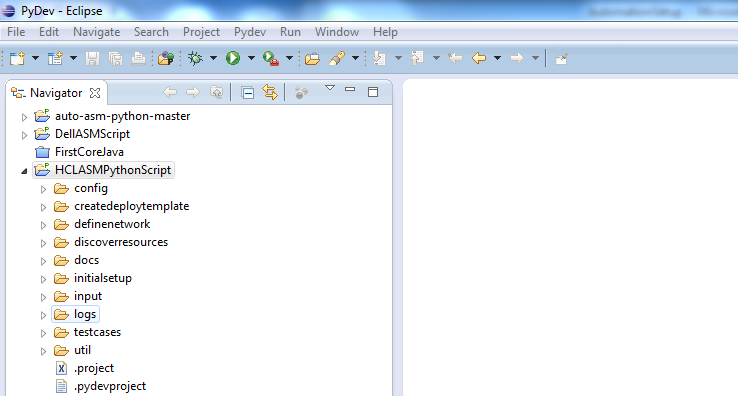
1. Open in the File → Import and Select the existing projects into Workspace.



1. Select the next and browse the project location.



1. Check all the checkboxes as shown in above and click on finish. The Imported project will be look like below



**Running a test case using python in eclipse**

1. Go to testcases folder , this folder contains three sub folders (common, esxi and hyperV)

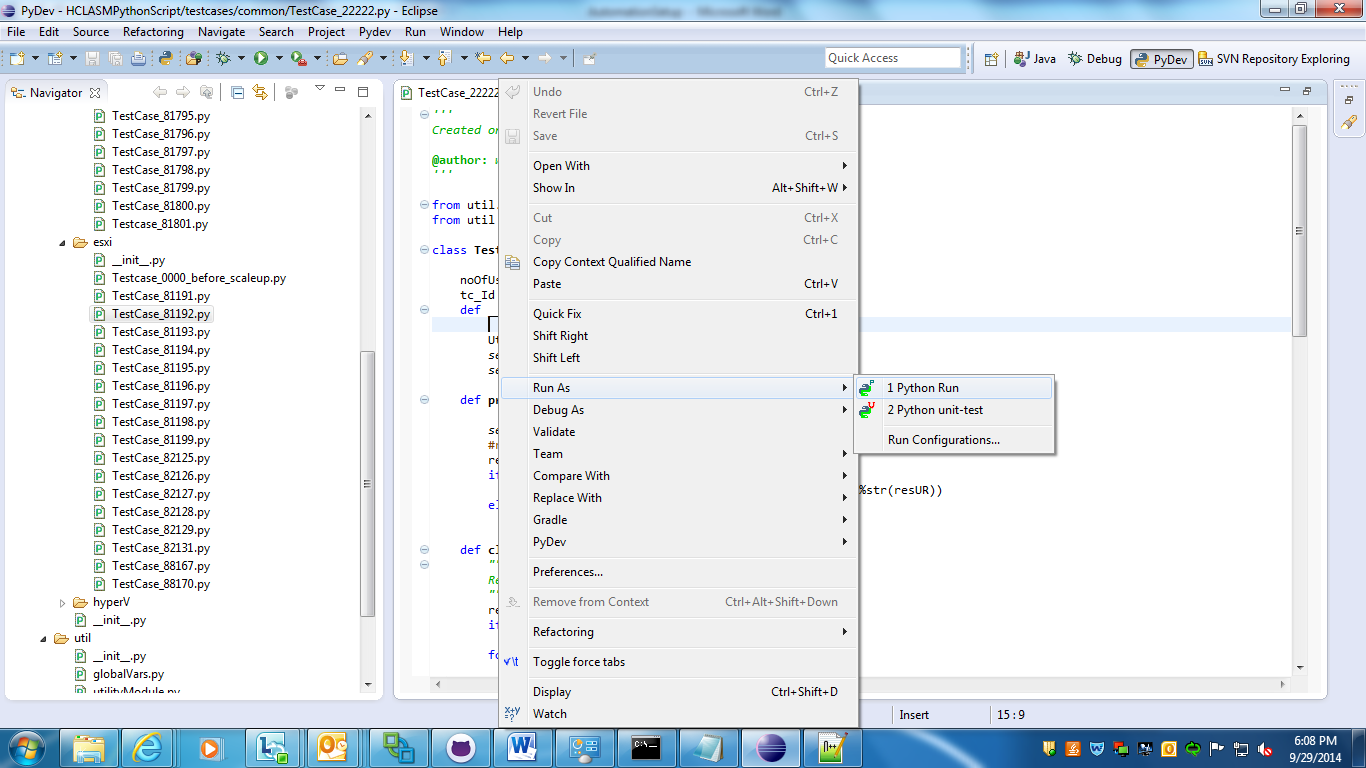
-> common : contains the common steps scripts like InitialSetup, DefineNetwork, discoverResources and ConfigureResources.

-> esxi : contains the scripts for esxi testcases

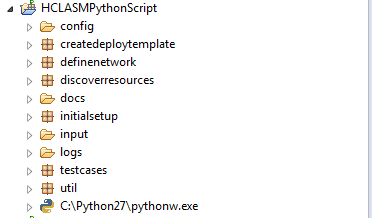
-> hyperV : contains the scripts for hyperV testcases

1. To run a test case, open the test case and right click on the test case and then run as python run.

Right Click → Run As → Python Run



**Project Folder Structure and their Details**



**config folder:** Config folder contains configuration and API's url related informations and it contains two files.

1. **config.ini:** it contains configuration information like Appliance IP and Port

**NOTE:** Add/modify configuration related info in this file (e.g. new Appliance IP/ Port)

1. **Services.xml:** It contains url information for different services of ASM. (e.g. Discovery url, ManageDevice url etc)

**docs folder:** docs folder contains the readme file.

**input folder:** This folder contains All the input information needed at different stages of Automation. This folder contains the below folders/files:

1. **payload Folder :** This folder contains the Discovery/Setup/ManageDevice related payload
2. **templates Folder:** This folder contains the payloads for Hyper-V testcases
3. **template\_esxi Folder:** This folder contains the payloads for ESXi testcases
4. **Credential.csv:** This file contains the credential related information for devices (Server, Storage, and Chassis, switch, scvmm and vcenter.)

**NOTE:** Change this file for any modification in credential related inputs.

1. **Network.csv:** This files Contains network related information for discovery of networks.

**NOTE**: change this file for any changes in network inputs

1. **Discovery.csv:** This files Contains the discovery of multiple resources **(servers/switches/Storage) where IP Addresses are not in range (we can give multiple resources to discover in one go)** Also resources can be discovered with different credentials.

**discovery.csv** file where we will be providing the information as below :

**SNo,Type,START\_IP,END\_IP,CredentialName,ChassisServerCre,ChassisSwitchCre**

1, SERVER,172.21.0.52,172.21.0.54,autoServer,,

2, CHASSIS,172.21.0.150,172.21.0.150,HVChasis,autoServer,autoIOM

3, SWITCH,172.21.10.248,172.21.10.248,autoSwitch,,

4, SWITCH,172.21.10.249,172.21.10.249,WSwitch,,

The columns **ChassisServerCre** and **ChassisSwitchCre** are the credentials for Servers and switches connected to a particular chassis.

Hence these fields are required/mandatory in case of Chassis whereas for other devices these fields will be blank.

This is done to make the Chassis discovery configurable.

**NOTE** : ***The CredentailName, ChassisServerCre amd ChassisSwitchCre should already be present in Credential.csv file.***

*These columns are actually subset of “Name” column in credential.csv file*..

**NOTE**: change this file for any changes in network inputs

**log folder:** This folder contains All the logs . This folder contains the below folders/files

**Note: Folder and files will be created when run the testcases.**

1. **testcase\_log Folder :** This folder contains the logs of each test cases and with the name of test case.
2. **ASMlogfile.log :** This file contains the log of all the operations.
3. **testcasesResultInfo.csv:** As the name suggested this contains that a testcase is passed are failed information.

**testcases folder:** This folder contains three sub folders (common, esxi and hyperV)

1. **common Folder:** This folder contains the common testcases.
2. **esxi Folder:** This folder contains the test cases related to the exsi.
3. **hyperV Folder:** This folder contains the test cases related to the hyperV.
4. **fcoe\_flex\_fc\_mxl Folder:** This folder contains the test cases related to the Flex FC in MXL.
5. **fcoe\_flexioa Folder:** This folder contains the test cases related to the Flex FC in IOA.
6. **fcoe\_nobrocade Folder:** This folder contains the test cases related to the Cisco Nexus without Brocade.
7. **fcoe\_s500withbrocade Folder:** This folder contains the test cases related to the s5000 with Brocade.
8. **fcoe\_s500withoutbrocade Folder:** This folder contains the test cases related to the s5000 without Brocade.
9. **fcoe\_withbrocade Folder:** This folder contains the test cases related to the Cisco Nexus with Brocade.
10. **firmwarerepository Folder:** This folder contains the test cases related to the OS and Frimware Repository creation.
11. **teardownresources Folder:** This folder contains the test cases related to the delete a device from successful deployment.
12. **chassisconfig Folder:** This folder contains the test cases related to the discover a chassis and configure the chassis.
13. is failed then Teardown the Deployed Service only.

**Regression/ Sanity Test Cases:**

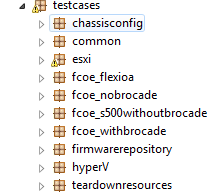
Each folder contains its own Regression case, Under the Regression we are doing following tasks in order:

1. Teardown all the Deployed Service
2. Teardown all the Published Templates
3. Teardown all the Server Polls.
4. Teardown all the Discovered Devices.
5. Teardown all the Defined Networks
6. Teardown all the Defined Credentials.
7. Teardown all the Os Repository.
8. Run the Initial Setup.
9. Define all the Networks Defined in Network.csv.
10. Define Credentials from **Credential.csv** and Discover all the Resources Defined in **Discovery.csv**
11. Run the configure Resource.
12. Create the Os Repository for different imageType, name and sourcePath.
13. For each testCaseId defined in the Regression testcase . it will repeat the steps no 14 to 18
14. Create Templates from the Defined testcase ID.
15. Publish Templates from the Defined testcase ID
16. For Published Template it runs the Deployment Service.
17. If Deployment Service is successful then Teardown the Deployed Service and delete Published Templates.
18. If Deployment Service is failed then Teardown the Deployed Service only.

**Process to Automate or Add new Test Cases:**

We have different folder based on the Testcase type like **esxi, hyperV and fcoe\_flex\_fc\_mxl etc.** Suppose we want to add a new test case for the esxi flow then we have to do the following steps to integrate the new test case in framework.

1. **Classify the test case**: First decide the new test case falls under which category in the testcases package. i.e.



1. **Develop the script** : While developing the python script firstly import the below python packages that will be used :

import json

import xml.etree.cElementTree as ET

import time

Also include the below packages/files of our framework :

from createdeploytemplate.TemplateBaseClass import TemplateTestBase

from util import globalVars

from discoverresources.DiscoverResourceBaseClass import DiscoverResourceTestBase

Include/import other packages required as per test case requirement.

1. **Prepare and place the payload:** The payload required for the test case should be placed inside input folder in the requisite subfolder i.e. payload,templates,templates\_esxi,templates\_flexIoa.

Payload file should be named as follows:

filename\_TestCase\_testcaseId.

1. **Provide the mapping of payload path in globalVars.py:**

In the python script the payload is read from the path of the payload file provided in globalVars.py.

e.g. filename\_TestCase\_77452=*"../../input/templates/TestCase\_77452.txt"*

1. **Run the script**: Currently we can run the individual script by doing a right click on the python script and selecting “ Run as python “

**Process to update existing Test Cases:** If a specific input value needs to be changed then we have multiple files where we need to change the value. PFB the list of files

1. **initialSetupReqValue.py:**  changed the NTP value in this file
2. **templateInputReqValue.py:** Changed the parameterized value of paylod
3. **inputReqValueEsxi.py:** This contains parameter value that get replaced in the the ESXi deployment template.
4. **Credential.csv:** This is an input file which contains information for defining credentials.
5. **Discovery.csv:** This file contains input information for resource discovery.
6. **Network.csv:** This file contains information for defining networks.
7. **TearDown.csv:** In this file we provide information for deleting resources.
8. **globalVars.py:** This file contains the absolute paths of all the input/template/testcases files or scripts. Also it contains value of some parameters that can change at run time.

**Note:** Before running the python script, Please follow the following steps:

1. Login in to the appliance and disable the firewall using below command
2. • sudo service iptables save

• sudo service iptables stop

• sudo chkconfig iptables off

**Steps to be performed for Firmware update automation:**

As part of firmware automation we perform the following tasks:

* 1. Create Firmware Repository
  2. Make the newly created repository as Default Repository.
  3. Wait until the Repository gets downloaded (There will be a message on the Eclipse console when the download is complete or Error message in case of any.)
  4. Finally the script checks what all devices are in “Non-Compliant” or “Update Required” state in the resources tab and updates the same.

The above tasks are integrated in the Regression suite as part of initial configuration/setup of appliance. The script that needs to be run is **TestCase\_ExsiRegression.py under the hierarchy : HCLASMPythonScript\testcases\esxi\**

Input fir the filepath to be choosen for Firmware Repository has to be provided in parameter file RepositoryParamFile.py which contains variables for nfs, cfs source locations.

**E.g. nfsSourceLocation =** r'\\10.255.7.219\SELab\LAB\Catalog\ASMBaselineTest\ASMCatalog.xml'

The above mentioned steps are part of a complete flow for a new ASM initial configuration.

However if the user wants to perform only Firmware Upgrade on an already configured ASM then below steps need to be done :

* + 1. Provide the path of the catalog in:

**Filename**: /HCLASMPythonScript/testcases/firmwarerepository/RepositoryParamFile.py

**Variable name** : nfsSourceLocation

**E**.g. : **nfsSourceLocation** = r'\\10.255.7.219\SELab\LAB\Catalog\ASMBaselineTest\ASMCatalog.xml'

* + 1. Gotoscript:/**HCLASMPythonScript/testcases/firmwarerepository/TestCase\_102640**.py and run this file to create firmware Repository
    2. Now run the below script to update firmware resources **/HCLASMPythonScript/testcases/firmwarerepository/FirmwareUpdate.py**

**Credentials for OS Repository creation :**

**To pass the credential in case of OS Repository we need to provide username and password in the parameter file : RepositoryParamFile.py in the below two variables :**

OSRepUserName = *""*

OSRepPasswd = *""*

*For e.g. :* OSRepUserName = Waseem.irshad

OSRepPasswd = \*\*\*\*\*\*\*\*

**Steps to update Qmetry from the python script:**

When will we execute a test case with python script, status of that testcase will be updated in Qmetry. But we have to follow the following steps before executing the testcases:

1. Go to the util folder and open the file QmetryInput.properties and update the value of following input parametes:
   1. QMETRY\_USERNAME = ProcessPGAutomation
   2. QMETRY\_PASSWORD = Qmetry@1234
   3. QMETRY\_PROJECT = ASM
   4. QMETRY\_RELEASE = 8.1
   5. QMETRY\_BUILD = Sprint 4
   6. TEST\_SUITE\_NAME = Testsuite\_FcoES500withbrocade
   7. PLATFORM\_NAME = FCoE
2. Login into Qmetry and create a Test Suite with the same name as TEST\_SUITE\_NAME.
3. Add all the testcases to Test Suite.
4. Link the Test Suite with a Platform with same name as PLATFORM\_NAME.

**Automation of Server Pool Creation for Regression script (Steps and logic)**

To create separate server pool for different flows for regression script follow below steps :

1. In globalVars.py we provide the name of the pool that we want and ip's of the servers that we want to be used for a particular pool

For e.g.

serverPoolNameExsi = "TestPool"

ServersIPForEsxiPool = ['0.0.0.0','1.1.1.1'....]

In the above example "TestPool" is the name of the pool that will be created containing servers with ips mentioned in "ServersIPForEsxiPool".

1. Similarly we can create separate pools for separate flows by providing name and server ip's.

LOGIC :

In Script TestCase\_RegressionAllCases.py function **createServerPools**(*self*,flow): is getting called before the start of every flow. The function takes the flow name as a parameter which is “globalVars.testCaseFlowName” mentioned in the script for each flow.

E.g. : globalVars.testCaseFlowName =*'esxi'*

test.createServerPools(globalVars.testCaseFlowName)

This function compares the flow name and gets the ip and pool name from globalVars corresponding to each flow.

E.g. : if flow == *'esxi'*:

poolName = globalVars.serverPoolNameExsi

ipList = globalVars.ServersIPForEsxiPool

Then it compares the ip given for pool creation with servers present in Resources and if found it appends the server information in a list.

E.g. : for device in *self*.responseMD:

for ips in ipList:

if device[*'ipAddress'*] == ips:

deviceList.append(device)

This list alongwith pool name is passed to the server pool creation function present in Discover Resources Base Class.

**self.createServerPool**(deviceList, poolName)