PyOne

For Rapid Automation

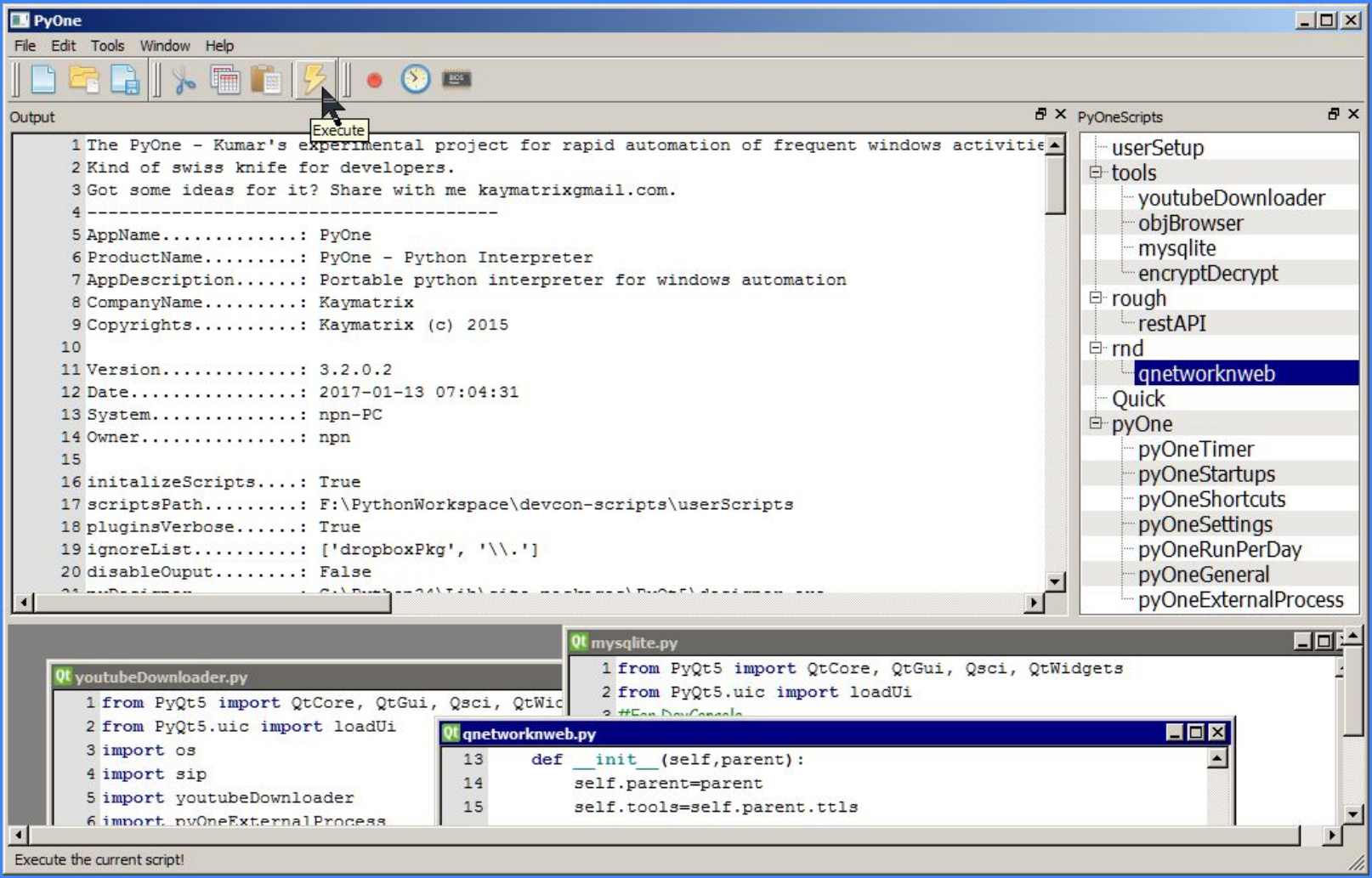
About PyOne

PyOne is a simple, portable, self-evolving - pyqt based python application developed for my personal and official use.

The main purpose of this tool is to do a rapid automation for any type of windows based development/testing and other activities, regardless of tools and technologies in existing place.

It self-evolves and supports to any situation, process, tools or systems for which users can build their own custom tools in amazing blazing speed.

All you need is very basic python knowledge.


PyOne
File Edit Tools Window Help
Execute
Output
1 The PyOne - Kumar's experimental project for rapid automation of frequent windows activitie
2 Kind of swiss knife for developers.
3 Got some ideas for it? Share with me kaymatrixgmail.com.
4 ----------
5 AppName.....: PyOne
6 ProductName.....: PyOne - Python Interpreter
7 AppDescription.....: Portable python interpreter for windows automation
8 CompanyName.....: Kaymatrix
9 Copyrights.....: Kaymatrix (c) 2015
10
11 Version.....: 3.2.0.2
12 Date.....: 2017-01-13 07:04:31
13 System.....: npn-PC
14 Owner.....: npn
15
16 initalizeScripts....: True
17 scriptsPath.....: F:\PythonWorkspace\devcon-scripts\userScripts
18 pluginsVerbose.....: True
19 ignoreList.....: ['dropboxPkg', '\\.']
20 disableOuput.....: False
PyOneScripts
userSetup
tools
youtubeDownloader
objBrowser
mysqlite
encryptDecrypt
rough
restAPI
rnd
qnetworknweb
Quick
pyOne
pyOneTimer
pyOneStartups
pyOneShortcuts
pyOneSettings
pyOneRunPerDay
pyOneGeneral
pyOneExternalProcess
youtubeDownloader.py
1 from PyQt5 import QtCore, QtGui, Qsci, QtWic
2 from PyQt5.uic import loadUi
3 import os
4 import sip
5 import youtubeDownloader
mysqlite.py
1 from PyQt5 import QtCore, QtGui, Qsci, QtWidgets
2 from PyQt5.uic import loadUi
qnetworknweb.py
13 def __init__(self,parent):
14 self.parent=parent
15 self.tools=self.parent.ttls
Execute the current script!

PyOne's Interesting challenges

A Manual deployment has a series of steps like Editing XMLs, File updates, CSV updates, cross checking server names etc. Which should be automated with visually clean neat GUI so users can simply click items and buttons.

Daily shakeout testing fetches details about various server health and serve the reports in a custom web service hosted with-in, so people can check reports from remote.

And many more… Explore yourself.

Daily check IND/AUD Currency rate and send me alert when threshold level reached. also log the rates to DB and give me a beautiful graph.

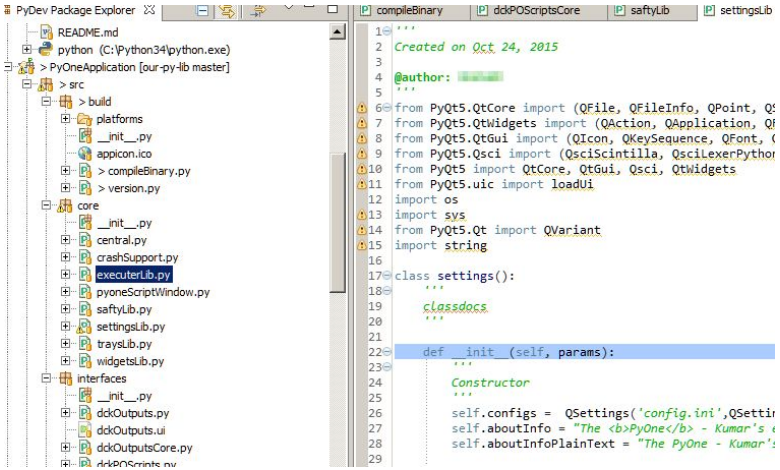
Call a specific RESTful service and fetch JSON details, Based on it cross check data in my Google Cloud and update the file present in my Dropbox Cloud.

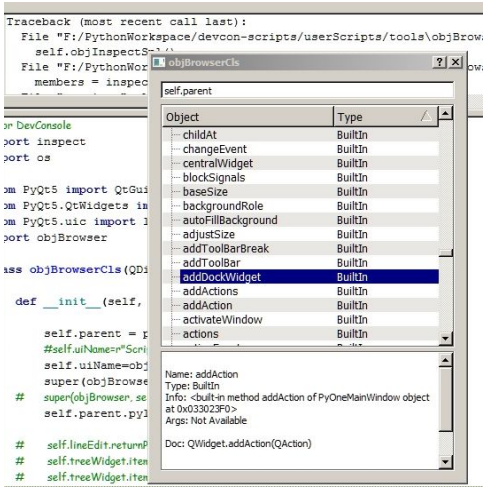
Firefighter Situation: 50k DB Records need to be manually checked against 10k xml file data. Based on result need to update various other xml files. All need to be done in 2 days.

More About PyOne

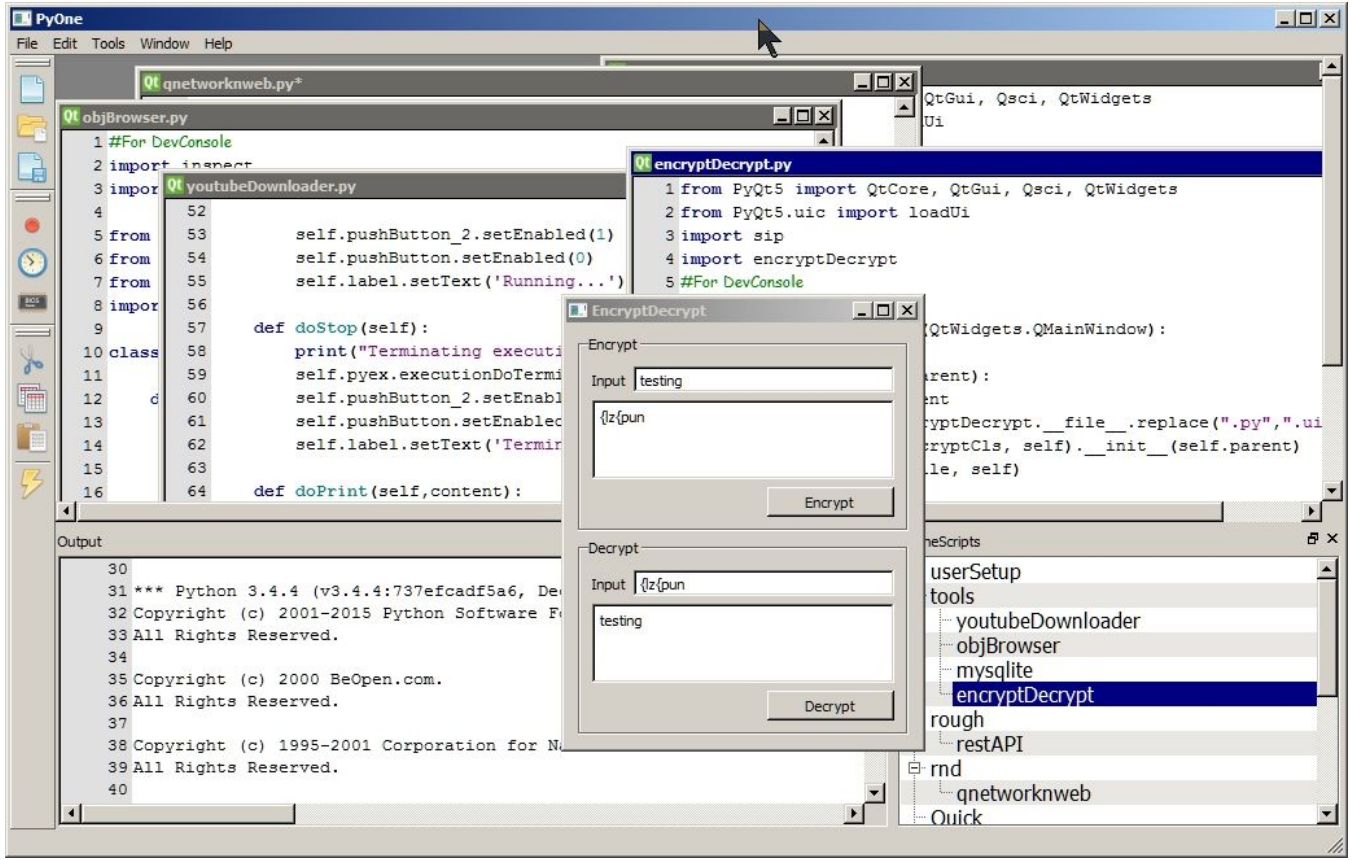
PyOne Build

- Built is based upon x86 – 64bit Python 3.0 with PyQt5 and Other basic modules built in it.
- Compiled to Standard Standalone 64bit Binary executable for windows execution. (No Installation needed)
- PyOne has built-in Qt Based Python Interactive Interpreter with all most all PyQt5 modules.
- PyOne uses PyQt GUI Designer (packed along with PyOne) for rapid GUI based tool building.
- More modules like mat-lab, graph plotting, pysci, pil, pygames etc can be included (Through source build).
- Entire application is openly available for rebuilding with-in itself. User has full freedom to transform the tool to their need.

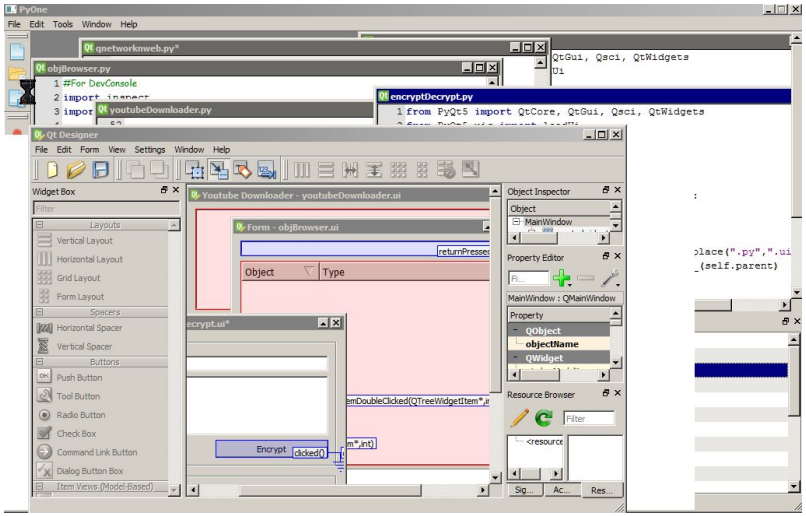


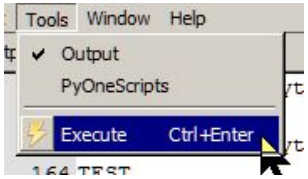


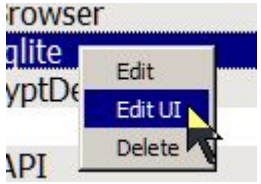
Flexible Framework

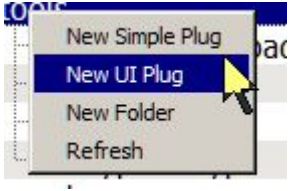


PyOne Framework

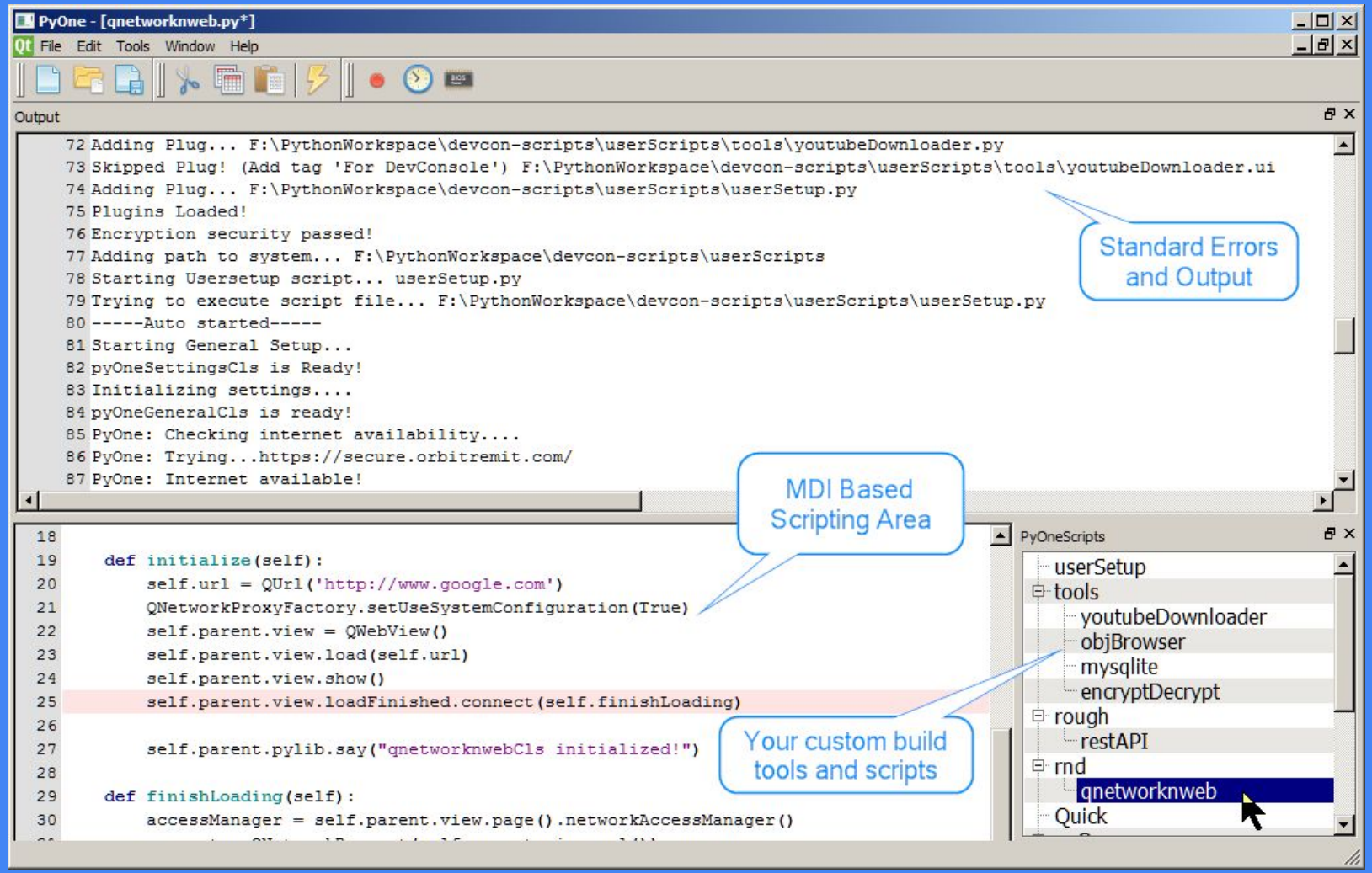






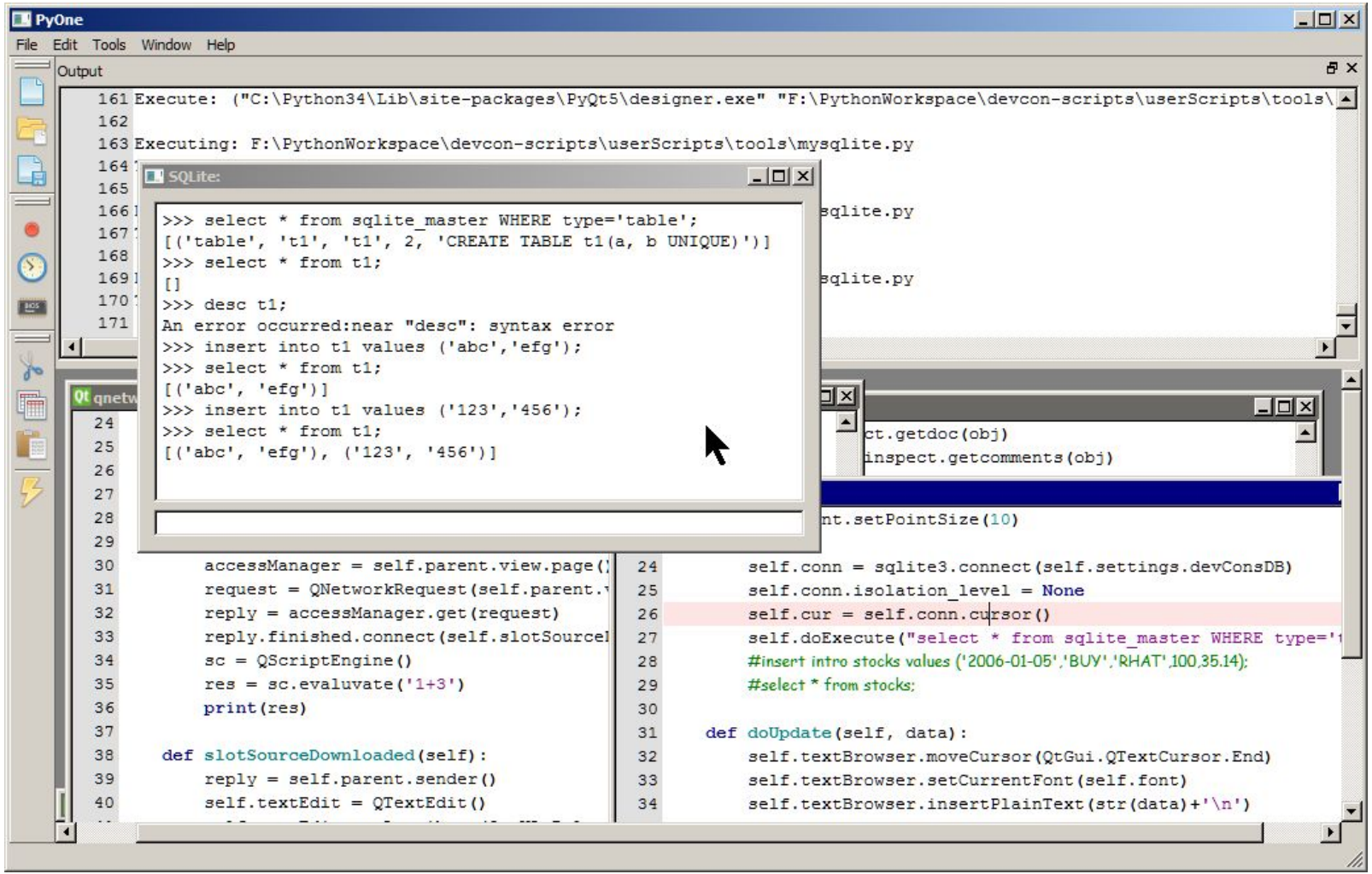


- PyOne scripting framework has nothing specific. It's almost similar to simple dynamic python execution.
- Framework has been defined with various structures for building the new plugins to it, which simple python users can use it with no time of learning.
- PyOne itself already evolved to a certain stage with help of its own Script Framework.
- GUI or NON-GUI any application can be built with this framework. GUIs are PyQt based loads dynamically.
- Built-in colourful icons (fatcows - farmfresh) for building your GUIs rapidly with colourful icons.
- MDI Editor, Editor enabled with QsciScintilla & Python Lexer, Dockable windows, Standard Error and Outputs, Dynamic Execution, Error Handling, Threads etc


PyOne - [qnetworknweb.py*]
File Edit Tools Window Help
Output
72 Adding Plug... F:\PythonWorkspace\devcon-scripts\userScripts\tools\youtubeDownloader.py
73 Skipped Plug! (Add tag 'For DevConsole') F:\PythonWorkspace\devcon-scripts\userScripts\tools\youtubeDownloader.ui
74 Adding Plug... F:\PythonWorkspace\devcon-scripts\userScripts\userSetup.py
75 Plugins Loaded!
76 Encryption security passed!
77 Adding path to system... F:\PythonWorkspace\devcon-scripts\userScripts
78 Starting Usersetup script... userSetup.py
79 Trying to execute script file... F:\PythonWorkspace\devcon-scripts\userScripts\userSetup.py
80 -----Auto started-----
81 Starting General Setup...
82 pyOneSettingsCls is Ready!
83 Initializing settings....
84 pyOneGeneralCls is ready!
85 PyOne: Checking internet availability....
86 PyOne: Trying...https://secure.orbitremit.com/
87 PyOne: Internet available!
Standard Errors and Output
MDI Based Scripting Area
18
19 def initialize(self):
20 self.url = QUrl('http://www.google.com')
21 QNetworkProxyFactory.setUseSystemConfiguration(True)
22 self.parent.view = QWebView()
23 self.parent.view.load(self.url)
24 self.parent.view.show()
25 self.parent.view.loadFinished.connect(self.finishLoading)
26
27 self.parent.pylib.say("qnetworknwebCls initialized!")
28
29 def finishLoading(self):
30 accessManager = self.parent.view.page().networkAccessManager()
Your custom build tools and scripts
PyOneScripts
userSetup
tools
youtubeDownloader
objBrowser
mysqlite
encryptDecrypt
rough
restAPI
rnd
qnetworknweb
Quick

Some more... Challenges for PyOne.

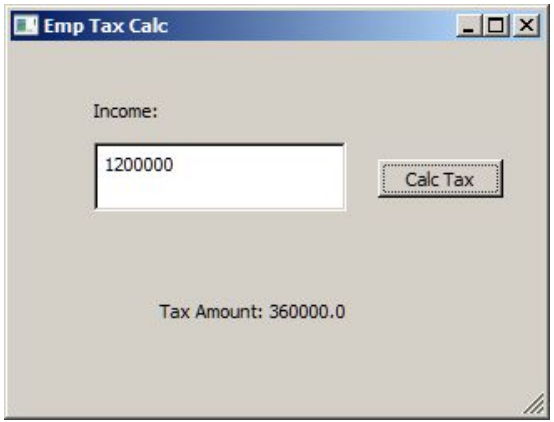
- Call a webservice in a specific interval and parse XML and convert to JSON and call RESTful.
- Quickly deploy/host a web service, handling the various GET/POST request. Also act as an application server..
- Download any youtube video based on viewer rating, comments, date, playlist, video formats etc.
- Perform and maintain health check data in SQL database of a various servers.
- Handle XML/XSD/XSL/HTML/JSON/CSV/TEXT/BATCH/BASH or any ASCII based file operations.
- Handles HTTPS/HTTP/SSL/SOAP/REST/Webservices with basic python modules.
- Various file-based operations like read/write/copy/delete/move/upload/download/sync files in specific interval or on-demand or based on result of various other activities.
- Encrypt/Decrypt base64 with custom keyword based security.
- Execute certain code daily once when the application starts.
- Execute certain code in specific interval of time.
- Execute custom DOS batch application on specific time and fetches the system-outs.
- Execute EXEs as visually customized shortcut cuts.
- And many more tools on the fly.


PyOne
File Edit Tools Window Help
Output
Execute: ("C:\Python34\Lib\site-packages\PyQt5\designer.exe" "F:\PythonWorkspace\devcon-scripts\userScripts\tools\
Executing: F:\PythonWorkspace\devcon-scripts\userScripts\tools\mysqlite.py
SQLite:
>>> select * from sqlite_master WHERE type='table';
[('table', 't1', 't1', 2, 'CREATE TABLE t1(a, b UNIQUE)')]
>>> select * from t1;
[]
>>> desc t1;
An error occurred:near "desc": syntax error
>>> insert into t1 values ('abc','efg');
>>> select * from t1;
[('abc', 'efg')]
>>> insert into t1 values ('123','456');
>>> select * from t1;
[('abc', 'efg'), ('123', '456')]
accessManager = self.parent.view.page()
request = QNetworkRequest(self.parent.
reply = accessManager.get(request)
reply.finished.connect(self.slotSourcel
sc = QScriptEngine()
res = sc.evaluate('1+3')
print(res)
def slotSourceDownloaded(self):
reply = self.parent.sender()
self.textEdit = QTextEdit()
self.conn = sqlite3.connect(self.settings.devConsDB)
self.conn.isolation_level = None
self.cur = self.conn.cursor()
self.doExecute("select * from sqlite_master WHERE type='
#insert intro stocks values ('2006-01-05','BUY','RHAT',100,35.14);
#select * from stocks;
def doUpdate(self, data):
self.textBrowser.moveCursor(QtGui.QTextCursor.End)
self.textBrowser.setCurrentFont(self.font)
self.textBrowser.insertPlainText(str(data)+'\n')

How to do?

A Simple tool in 10 mins

Let's quickly build a simple tool - Employee Tax Calc



Points to Consider

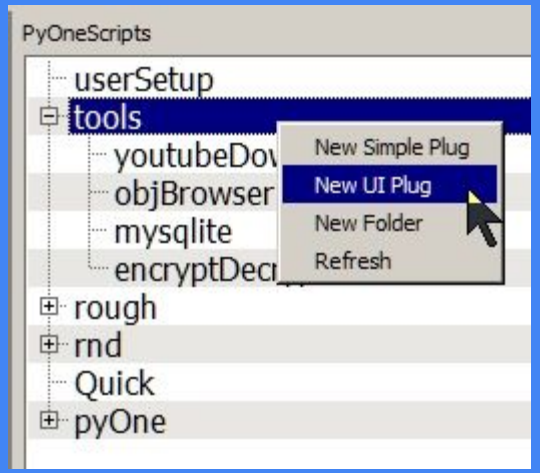
1. Should get user input Income amount from in a Text box.
2. Based on latest TAX Formula, calculate the TAX amount for the Income specified.
3. Display the TAX amount in another text area.
4. Calculation should happen on Click of a button.
5. And you got 10 mins time to do the tool.

Step 1

Right click any section under ‘PyOneScripts’

And

Select ‘New UI Plug’ Option

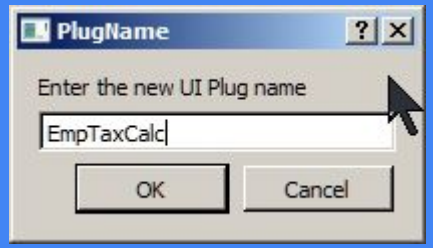


Step 2

Provide a name (EmpTaxCalc) for your new tool.

And

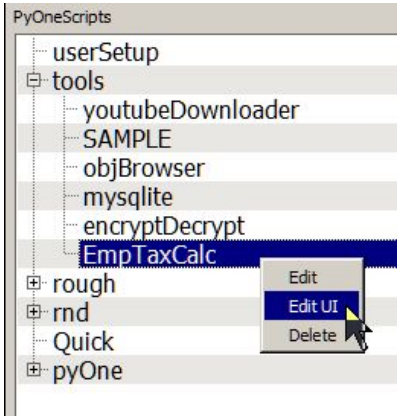
Click ‘OK’



Step 3

TaDa!!!
You are given with basic script and UI for editing and enhancing as per your requirement.

Now right click EmpTaxCalc in PyOneScripts and Click EditUI



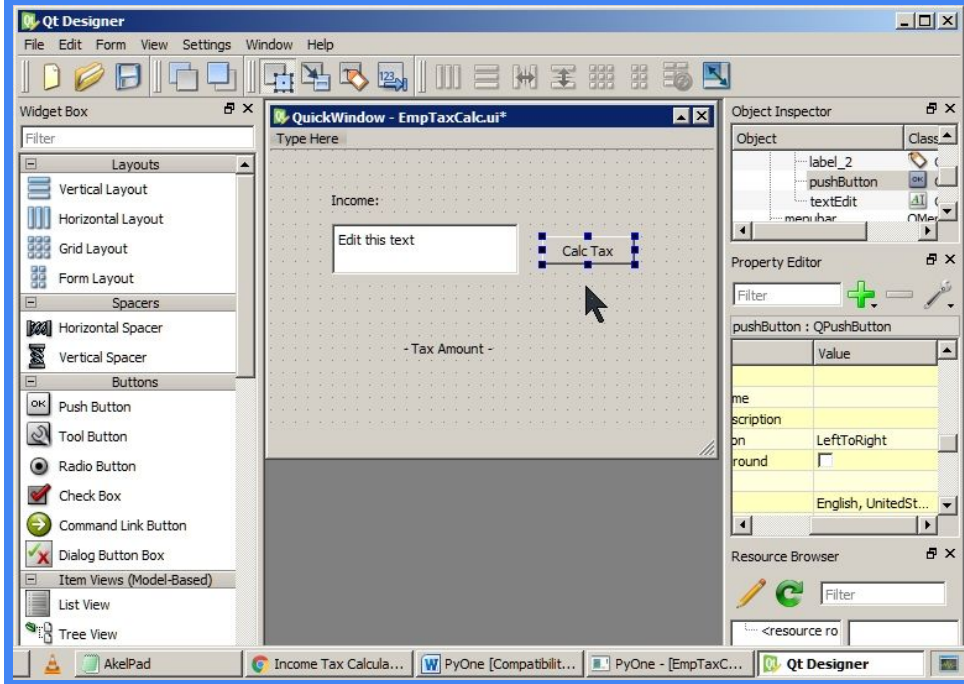
EmpTaxCalc.py

```
1  from PyQt5 import QtCore, QtGui, Qsci, QtWidgets
2  from PyQt5.uic import loadUi
3  import os
4  import sip
5  import EmpTaxCalc
6  #For DevConsole
7
8  class EmpTaxCalcCls(QtWidgets.QMainWindow):
9
10     def __init__(self,parent):
11         self.parent=parent
12         self.tools=self.parent.ttls
13         self.qtTools=self.parent.qtTools
14         self.uiFile=EmpTaxCalc.__file__.replace(".py",".ui")
15         super(EmpTaxCalcCls, self).__init__(self.parent)
16         loadUi(self.uiFile, self)
17         self.pushButton.clicked.connect(self.doRun)
18
19     def initialize(self):
20         self.parent.pylib.say("EmpTaxCalcClsObj is working fine")
21
22     def doRun(self):
```

Step 4

QtDesigner will be launched automatically with UI Script file.

Drag n Drop and Do your own GUI with the help of QtDesigner and save the file once you completed.

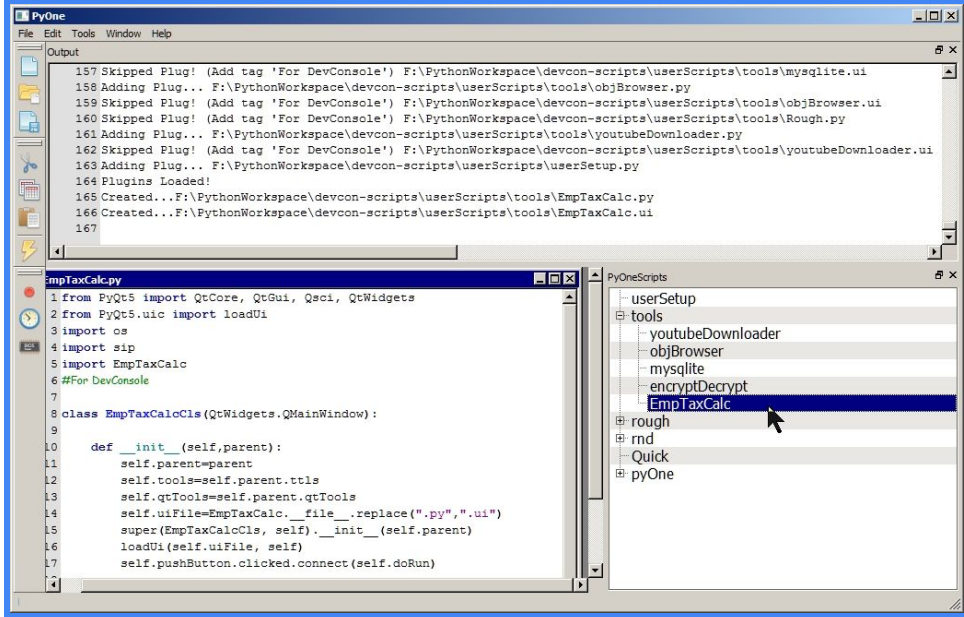


Step 6

Back in our PyOne Editor with EmpTaxCalc.py

Scroll down and look for a python method *doRun* as given below

```
def doRun(self):
    input = self.textEdit.toPlainText()
    self.label.setText(input)
    self.parent.pylib.say(input)
```



Step 7

Edit the doRun with below code and Save the script

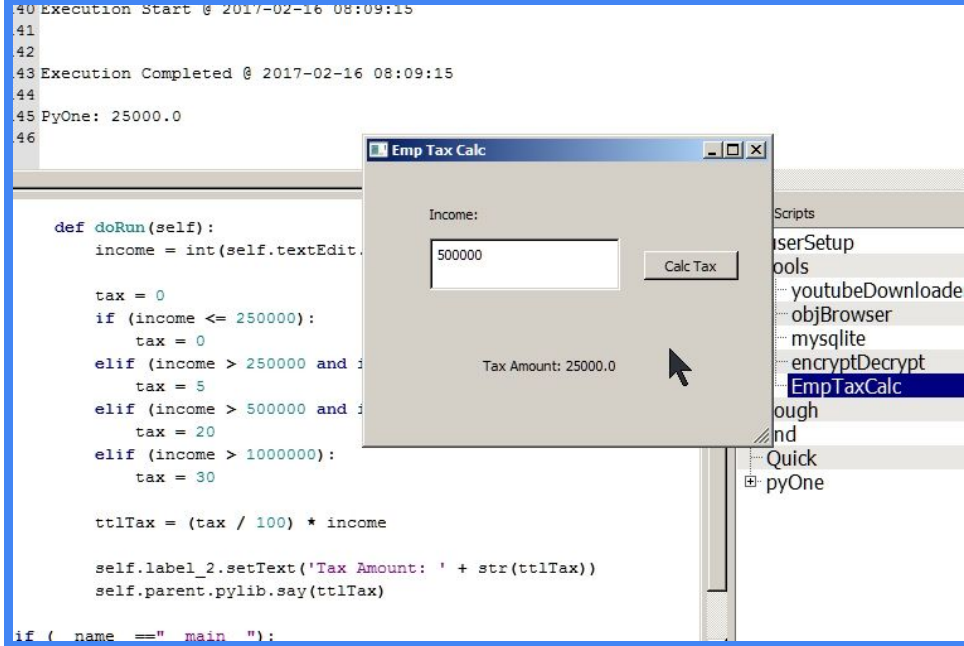
```
def doRun(self):
    income = int(self.textEdit.toPlainText())

    tax = 0
    if (income <= 250000):
        tax = 0
    elif (income > 250000 and income <= 500000):
        tax = 5
    elif (income > 500000 and income <= 1000000):
        tax = 20
    elif (income > 1000000):
        tax = 30

    ttlTax = (tax / 100) * income

    self.label_2.setText('Tax Amount: ' + str(ttlTax))
    self.parent.pylib.say(ttlTax)
```

Step 8

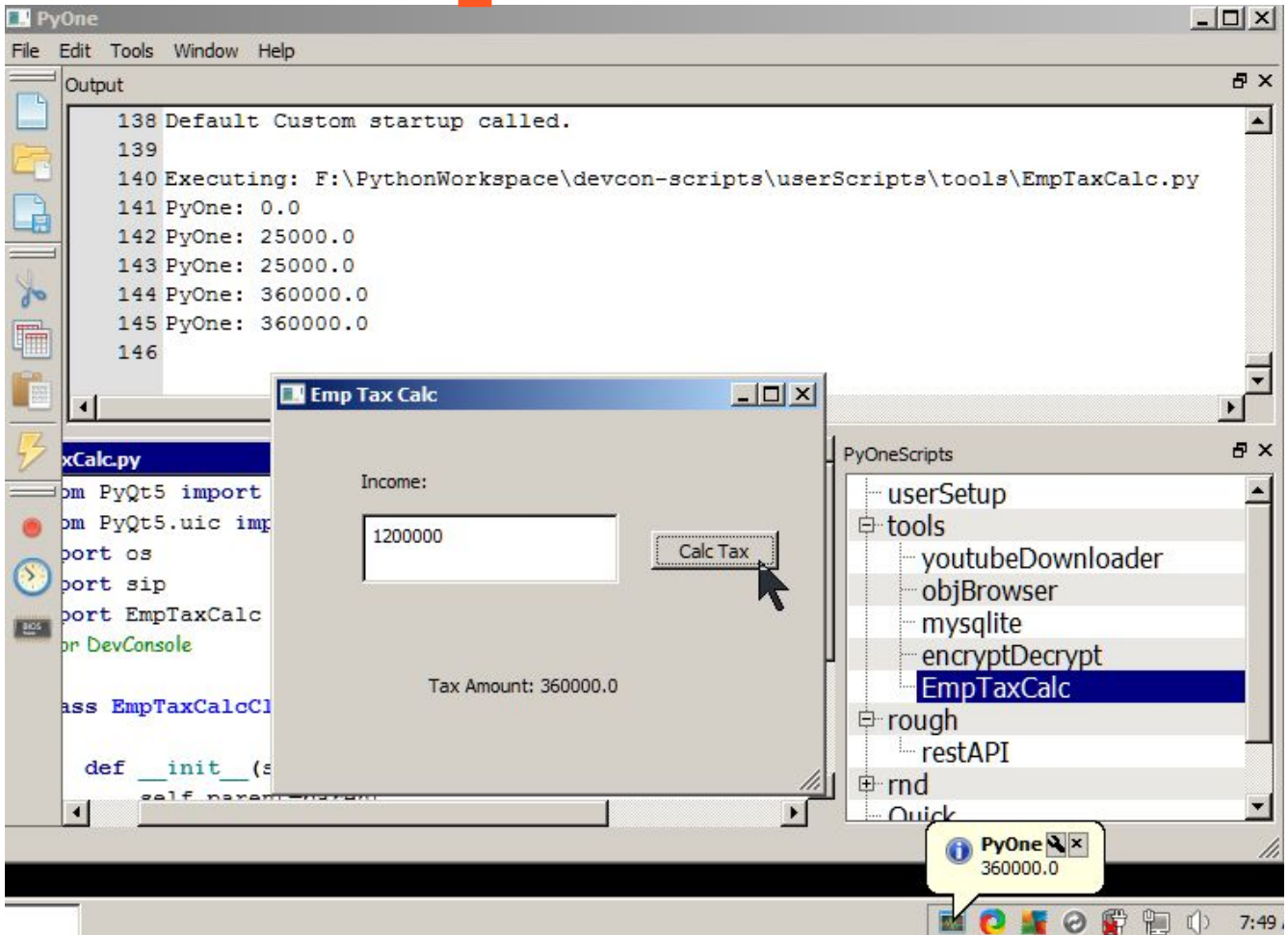
Execute the code by double clicking the EmpTaxCalc in PyOneScripts

And

BINGO!
Your EmpTaxCalc tool is done!



EmpTaxCalc


PyOne
File Edit Tools Window Help
Output
138 Default Custom startup called.
139
140 Executing: F:\PythonWorkspace\devcon-scripts\userScripts\tools\EmpTaxCalc.py
141 PyOne: 0.0
142 PyOne: 25000.0
143 PyOne: 25000.0
144 PyOne: 360000.0
145 PyOne: 360000.0
146
Emp Tax Calc
Income:
1200000
Calc Tax
Tax Amount: 360000.0
PyOneScripts
userSetup
tools
youtubeDownloader
objBrowser
mysqlite
encryptDecrypt
EmpTaxCalc
rough
restAPI
rnd
PyOne
360000.0
7:49

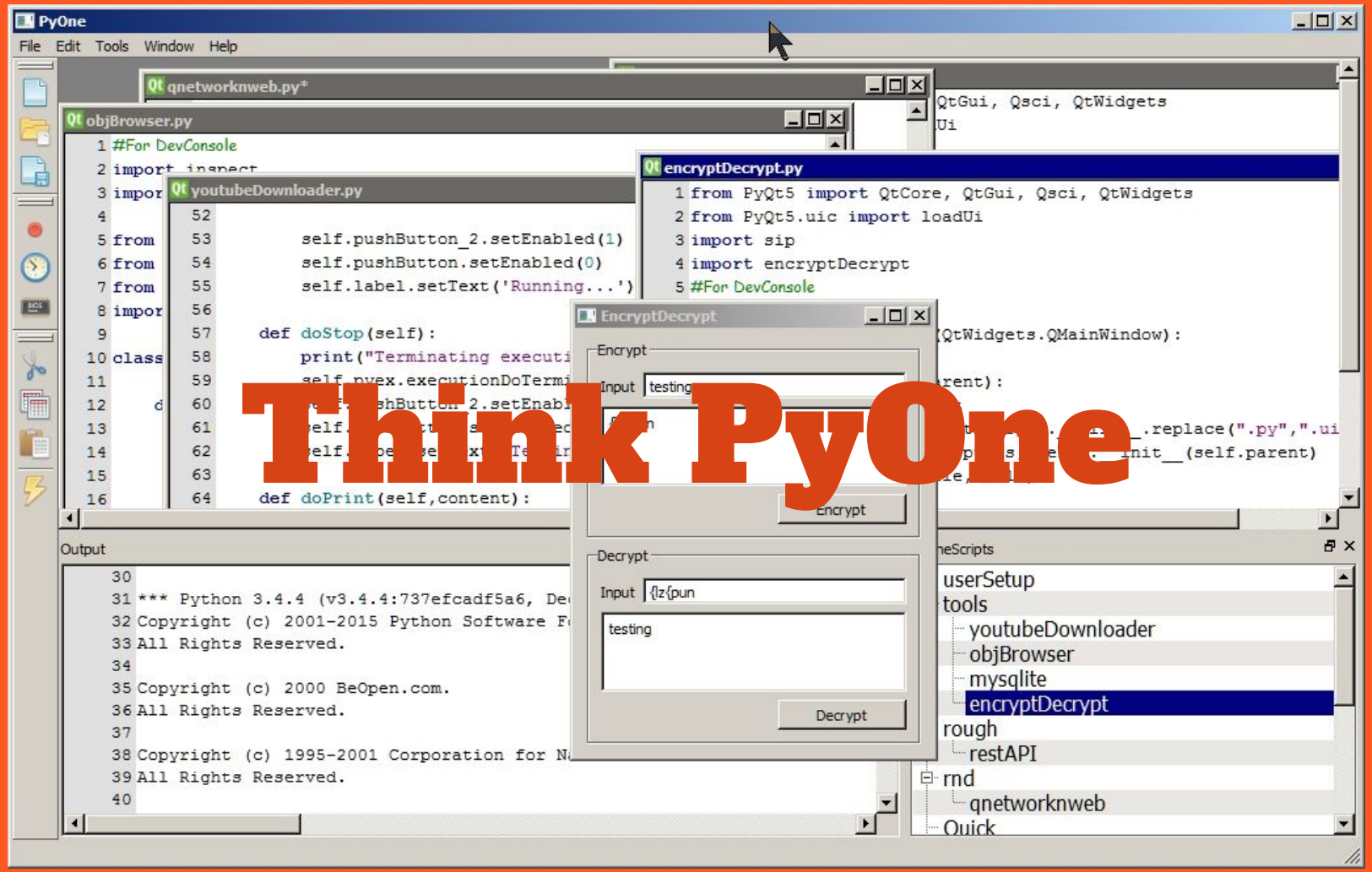
On seeing above sample, You may ask…

“What’s the big deal? I can do the same in MS-EXCEL or MS-CALC”.

Consider I change the challenge as given below…

1. TAX Rule is dynamic and available in CSV somewhere in web-server.
2. Need to do the same calculation for 5 lakhs employees fetching the income detail from a DB and prepare another CSV with all TAX calculated.
3. Need to repeat the calculation in specific interval.
4. Update Cloud with details processed.
5. With additional lot more, Challenges!!!

Think PyOne


PyOne
File Edit Tools Window Help
qnetworknweb.py*
objBrowser.py
youtubeDownloader.py
encryptDecrypt.py
EncryptDecrypt
Encrypt
Input testing
Encrypt
Decrypt
Input {\z{pun
testing
Decrypt
Output
userSetup
tools
youtubeDownloader
objBrowser
mysqlite
encryptDecrypt
rough
restAPI
rnd
qnetworknweb
Quick
Think PyOne

Disclaimer

PyOne features and functionalities are bound to the limitation of python, pyqt and windows. Most features and ability mentioned in this document are achieved with support of few third parties. It does exist few limitations to PyOne by its own which should be understood by users and can overcome with various third parties. PyOne author won't be responsible for any damages to hardware/ software or data loss caused by PyOne on mishandling them by the user. Please, discuss with author for gaining more knowledge about it. On critical usage.

\- PyOne Author

Note from author

I am Kumaresan Lakshmanan, author of PyOne Application. I am a mainstream developer (Java/Python/BPM/Integration etc) and I am not into any Devops. But still I hate doing redundant and manual works. Most time I used think of automation possibilities in my daily task like everyone does. But thought of having a customized unified portable solution. Thus I got my favorite programming language python to help me.

PyOne's key points are Rapid, Dynamic, Flexible and Simple. I got this inspiration from python itself. Also I love coding in python. It's been 7 years now. I explored lot many technologies, but I never felt anything as easy and enjoyable like python coding. So this is just a small contribution to my favourite python to show its flexibility and features to this world.

This tools is not meant for any commercial use. This will remain open-source in public github and users are free to fork and contribute to source.

Kumaresan Lakshmanan
Year 2017
Chennai - India

Thank you

Source Code

https://github.com/kaymatrix/our-py-lib/tree/master/PyOneApplication

Contact

kaymatrix@gmail.com

PyOne