



Macros Well Kept Secrets

LibreOfficiant

Macros Scripting Framework

→ **Python**

in 2008

→ **Basic**

*Office legacy

→ **API**

BeanShell

JavaScript

C++

Java

Python

“

”

Python

- ✓ Source explorer
- ✓ Syntax Highlighting
- ✓ Code completion
- ✓ Debugging / REPL
- ✓ Coding guidelines
PEP
- ✓ Test-driven
development
- ✓ Version control

→ Extensions

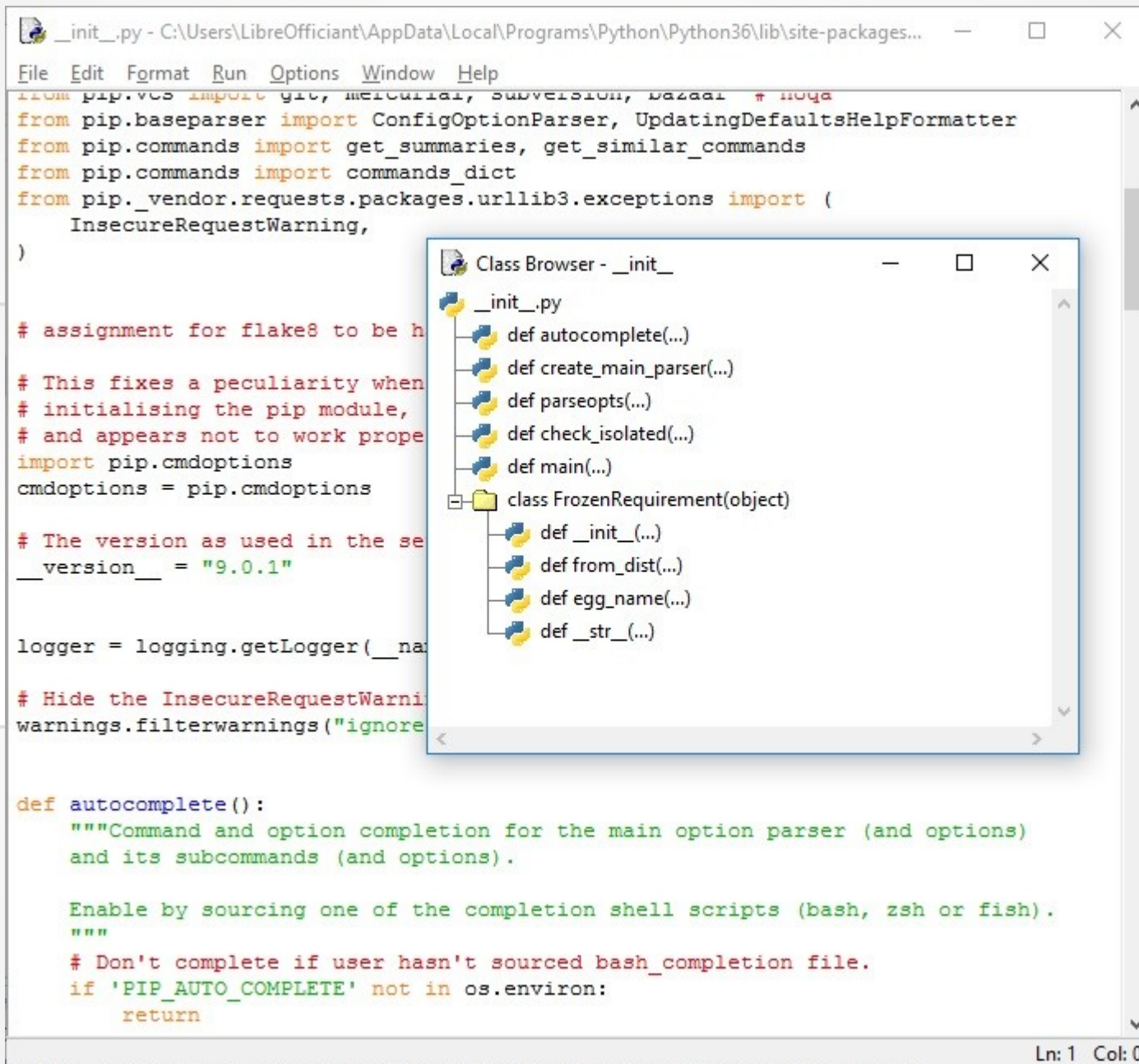
- ◆ Mri, Xray
- ◆ APSO

→ IDE's

- Python shell REPL
- IDE_utils
- Geany, PyZo,
PyCharm

“ ”

IDLE: class browser syntax highlighting



The screenshot shows the IDLE Python IDE interface. The main window displays a Python file named `_init_.py` with syntax highlighting. The code includes imports from `pip` and `requests`, followed by comments and assignments. A `def autocomplete():` function is defined at the bottom. A `Class Browser - _init_` window is overlaid on the main window, showing a tree view of the code's structure. The tree view includes the `_init_.py` file, several functions (`def autocomplete(...)`, `def create_main_parser(...)`, `def parseopts(...)`, `def check_isolated(...)`, `def main(...)`), and a class `class FrozenRequirement(object)` with its methods (`def __init__(...)`, `def from_dist(...)`, `def egg_name(...)`, `def __str__(...)`). The status bar at the bottom right indicates `Ln: 1 Col: 0`.

```
from pip.vcs import git, mercurial, subversion, bazaar
from pip.baseparser import ConfigOptionParser, UpdatingDefaultsHelpFormatter
from pip.commands import get_summaries, get_similar_commands
from pip.commands import commands_dict
from pip._vendor.requests.packages.urllib3.exceptions import (
    InsecureRequestWarning,
)

# assignment for flake8 to be happy
# This fixes a peculiarity when
# initialising the pip module,
# and appears not to work properly
import pip.cmdoptions
cmdoptions = pip.cmdoptions

# The version as used in the setup.py
__version__ = "9.0.1"

logger = logging.getLogger(__name__)

# Hide the InsecureRequestWarning
warnings.filterwarnings("ignore")

def autocomplete():
    """Command and option completion for the main option parser (and options)
    and its subcommands (and options).

    Enable by sourcing one of the completion shell scripts (bash, zsh or fish).
    """
    # Don't complete if user hasn't sourced bash_completion file.
    if 'PIP_AUTO_COMPLETE' not in os.environ:
        return
```

“

”

Geany: srce explorer, syntax hiliting, code folding

The screenshot displays the Geany IDE interface. The title bar shows the file path: `C:\Users\LibreOfficiant\Dropbox\LibreOffice 5\Apso\src - [LibreOffice] - Geany`. The menu bar includes: `Fichier`, `Éditer`, `Rechercher`, `Affichage`, `Document`, `Projet`, `Construire`, `Outils`, `Aide`. The toolbar contains icons for `Nouveau`, `Ouvrir`, `Enregistrer`, `Tout enregistrer`, `Rétablir`, `Fermer`, `Précédent`, `Suivant`, `Compiler`, `Construire`, `Exécuter`, and `Sélecteur de couleur`. The main editor area shows a Python script with syntax highlighting and code folding. The left sidebar displays a class explorer with a tree view of classes and methods.

```
1 import uno
2 import unohelper
3 import traceback
4 import sys
5
6 try:
7     import pythonscript
8 except:
9     import pythonloader
10    pythonscript = None
11    for url, module in pythonloader.g_loadedComponents.iteritems():
12        if url.endswith("script-provider-for-python/pythonscript.py"):
13            pythonscript = module
14    if pythonscript is None:
15        raise Exception("Failed to find pythonscript module.")
16
17 from com.sun.star.awt import XActionListener, XMouseListener, \
18    XKeyListener, Rectangle, Selection
19 from com.sun.star.awt.tree import XTreeExpansionListener
20 from com.sun.star.uno import Exception as UNOException
21
22 # Default content of the new file.
23 TEMPLATE = """
24
25
26 class DialogBase(object):
27     """ Base class for dialog. """
28     def __init__(self, ctx):
29         self.ctx = ctx
30
31     def create(self, name):
32         """ Create service instance. """
33         return self.ctx.getServiceManager().createInstanceWithContext(
```

Class Explorer (Classes):

- DialogBase [26]
- ErrorAsMessage [287]
- ErrorMessageDialog [248]
- FileOpenDialog [178]
- MessageDialog [207]
- NameInput [134]
- NodeManager [292]
- OrganizerDialog [342]
- ActionListener [591]
 - BUTTON_HEIGHT [360]
 - BUTTON_WIDTH [359]
 - DEFAULT_NAME [350]
 - DIR_ICON [354]
 - DISK_ICON [352]
 - DOC_ICON [353]
 - DOC_PROTOCOL [348]
 - ENABLE_DEBUG [366]
 - ENABLE_EDIT [365]
 - FILE_EXT [347]
 - FILE_ICON [355]
 - HEIGHT [362]
 - KeyListener [5051]

Terminal:

```
20:54:46: Voici Geany 1.28.
20:54:46: Projet « LibreOffice » ouvert.
20:54:46: Fichier C:\Users\LibreOfficiant\Dropbox\LibreOffice 5\Apso\src\apso-0.8.7.py ouvert (1)
```

System Tray: 21:09 09/03/2017

PyCharm: various explorers, syntax highlighting, code folding

The screenshot displays the PyCharm IDE interface. On the left, the 'Structure' tool window shows a project tree for 'apso.py' with various classes and methods listed, such as 'ErrorMessage(Exception)', 'NodeManager(object)', and 'DialogBase(object)'. The main editor window shows the source code of 'apso.py' with syntax highlighting. A tooltip is visible over the 'DialogBase' class definition, listing its subclasses: 'NameInput', 'ErrorMessageDialog', 'SyntaxErrorMessageDialog', and 'OrganizerDialog'. The bottom status bar includes '6: TODO', 'Python Console', and 'Terminal'.

```
apso - [C:\Users\LibreOfficiant\PycharmProjects\apso] - ...\apso.py - PyCharm Community Edition 2016.3.2
File Edit View Navigate Code Refactor Run Tools VCS Window Help
apso > apso.py >
Structure
1: Project
2: Structure
3: Favorites
  C ErrorMessage(Exception)
  C ErrorAsMessage(ErrorMessage_)
  C ErrorAsMessage(Exception)
  C NodeManager(object)
  C OrganizerDialog(NodeManager, RuntimeDialogBase)
    m __init__(self, ctx, user_provider, share_provider, d
    m _get_current_document(self)
    m execute(self, history=None)
    m button_pushed(self, command)
    m exec_execute(self, tree_node, node)
    m exec_menu(self, tree_node, node)
    m exec_create_file(self, tree_node, node, filename=
    m exec_create_dir(self, tree_node, node)
    m exec_substitute(self, tree_node, node)
    m exec_rename(self, tree_node, node)
    m exec_copypodoc(self, tree_node, node)
    m exec_export(self, tree_node, node)
    m exec_delete(self, tree_node, node)
    m exec_edit(self, tree_node, node)
    m exec_debug(self, tree_node, node)
    m _create_tempfile(self, node)
  C ListenerBase(unohelper.Base)
  C ActionListener(ListenerBase, XActionListener)
  C KeyListener(ListenerBase, XKeyListener)
    m _key_pressed(self)
  C MouseListener(ListenerBase, XMouseListener)
    m _mouse_pressed(self, ev)
  C TreeExpansionListener(ListenerBase, XTreeExpan
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
def resolvestring(self, id):
    return self.srwl.resolveString(id)

def loadResourceResolver(ctx):
    global RR
    if not RR:
        RR = ResourceResolver(ctx)

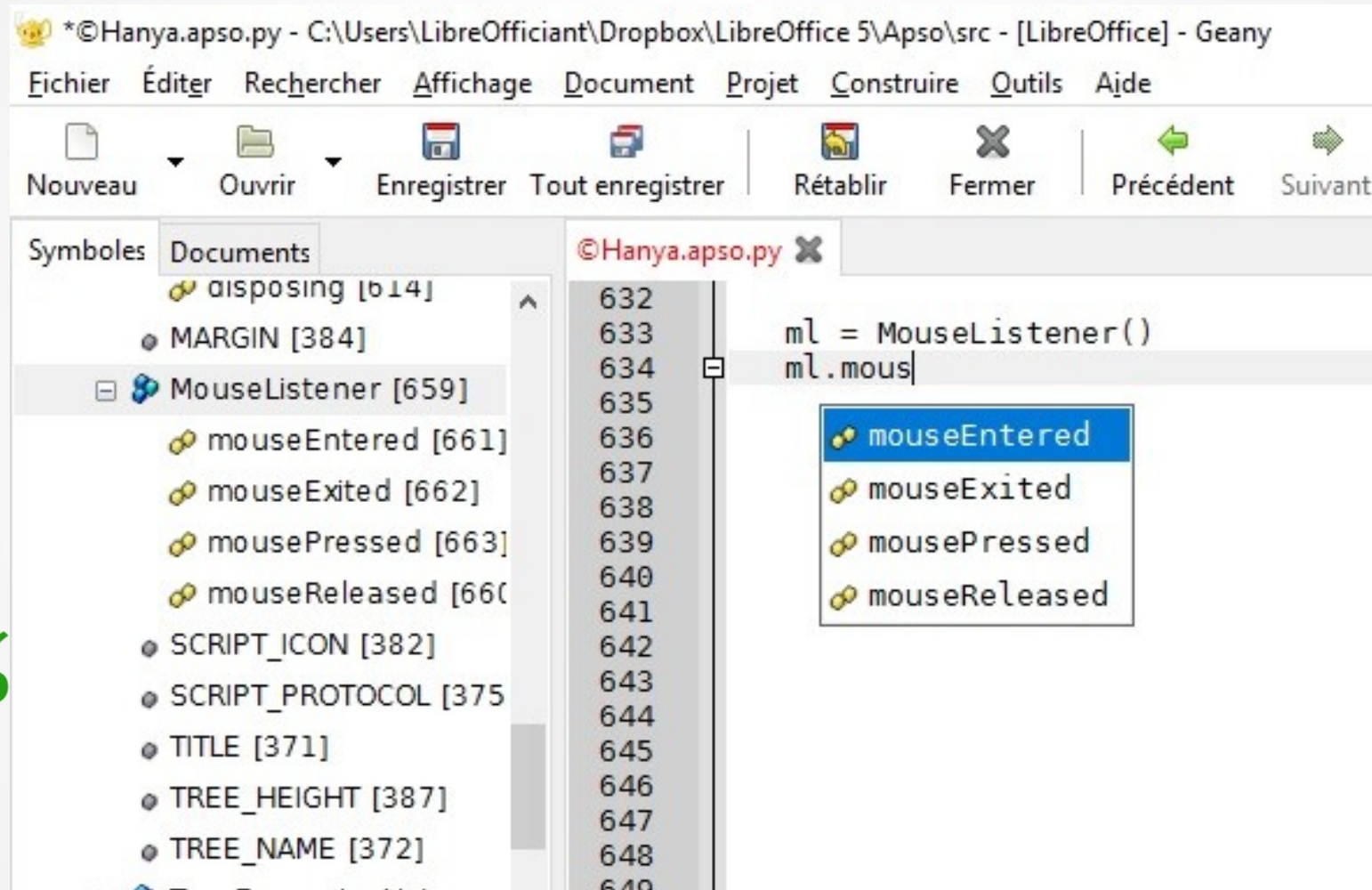
class DialogBase(object):
    """ Base class for dialog. """
    def __init__(self, ctx):
        self.ctx = ctx
        self.smgr = ctx.getServiceManager()

    def create(self, name, arguments=None):
        """ Create service instance. """
        if arguments:
            return self.smgr.createInstanceWithArgumentsAndCo
                name, arguments, self.ctx)
        else:
            return self.smgr.createInstanceWithContext(
                name, self.ctx)

    Is subclassed by:
    NameInput
    ErrorMessageDialog
    SyntaxErrorMessageDialog
    OrganizerDialog

    def __init__(self, ctx):
        DialogBase.__init__(self, ctx)
        self.dialog = None
```

Geany: code completion



PyCharm: code completion

PC apso - [C:\Users\LibreOfficiant\PycharmProjects\apso] - ...\apso.py - PyCharm Community Edition 2016.3.2

File Edit View Navigate Code Refactor Run Tools VCS Window Help

apso > apso.py >

Structure

- 1: Project
 - Structure
 - _mouse_pressed(self, ev)
 - TreeExpansionListener(ListenerBase, XTreeExpansionListener)
 - treeExpanding(self, ev)
 - treeCollapsing(self, ev)
 - treeExpanded(self, ev)
 - treeCollapsed(self, ev)
 - requestChildNodes(self, ev)
 - SelectionChangeListener(ListenerBase, XSelectionChangeListener)

apso.py x

```
616
617 tel = Tre
618
619
620
621
622
623
624
625
626
```

TreeExpansionListener OrganizerDialog
TREE_HEIGHT OrganizerDialog
TREE_NAME OrganizerDialog
XTreeExpansionListener
UnicodeTranslateError builtins

Press Ctrl+Point to choose the selected (or first) suggestion and insert a dot afterwards >> π

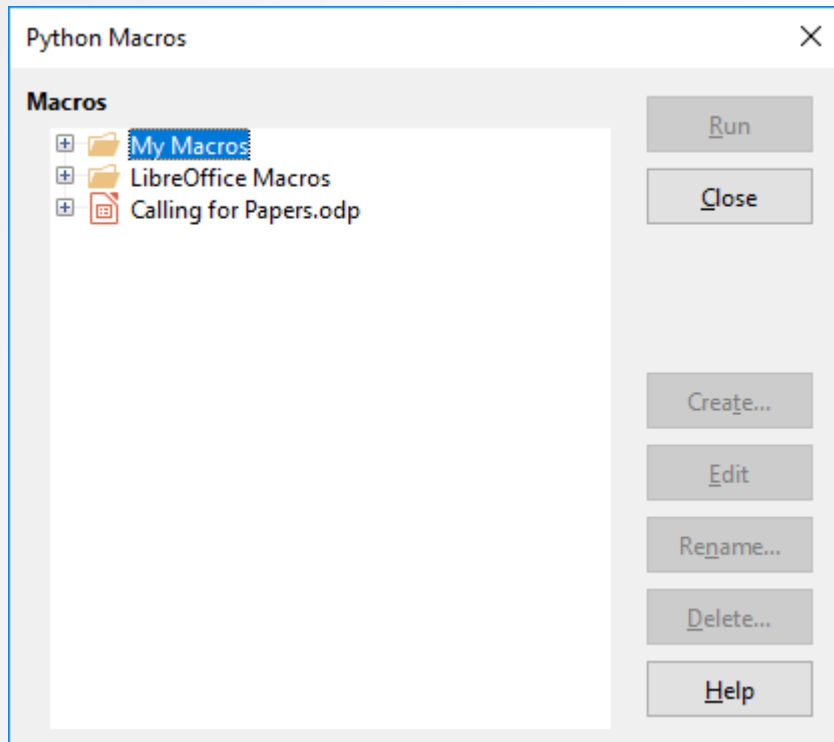
APSO: interactive console

```
APSO console
APSO python console [LibreOffice]
3.5.4 (default, May 9 2018, 21:39:23) [MSC v.1900 64 bit (AMD64)]
Type "help", "copyright", "credits" or "license" for more information.
>>> ctx = XSCRIPTCONTEXT.getComponentContext()
>>> smgr = ctx.ServiceManager
>>> geb = smgr.createInstanceWithContext("com.sun.star.frame.GlobalEventBroadcaster", ctx)
>>> dir(geb.Events)
['ElementNames', 'ElementType', 'Events', 'ImplementationId', 'OnCloseApp', 'OnCopyTo',
'OnCopyToDone', 'OnCopyToFailed', 'OnCreate', 'OnFocus', 'OnLoad', 'OnLoadFinished',
'OnModeChanged', 'OnModifyChanged', 'OnNew', 'OnPrepareUnload', 'OnPrepareViewClosing', 'OnPrint',
'OnSave', 'OnSaveAs', 'OnSaveAsDone', 'OnSaveAsFailed', 'OnSaveDone', 'OnSaveFailed',
'OnStartApp', 'OnStorageChanged', 'OnTitleChanged', 'OnUnfocus', 'OnUnload', 'OnViewClosed',
'OnViewCreated', 'OnVisAreaChanged', 'Types', 'getByName', 'getElementNames', 'getElementType',
'getEvents', 'getImplementationId', 'getTypes', 'hasByName', 'hasElements', 'queryAdapter',
'queryInterface', 'replaceByName']
>>> help(geb.Events.getElementNames)
Help on PyUNO_callable object:

class PyUNO_callable(object)
|   Methods defined here:
|
|   __call__(self, /, *args, **kwargs)
|       Call self as a function.
>>> |
```

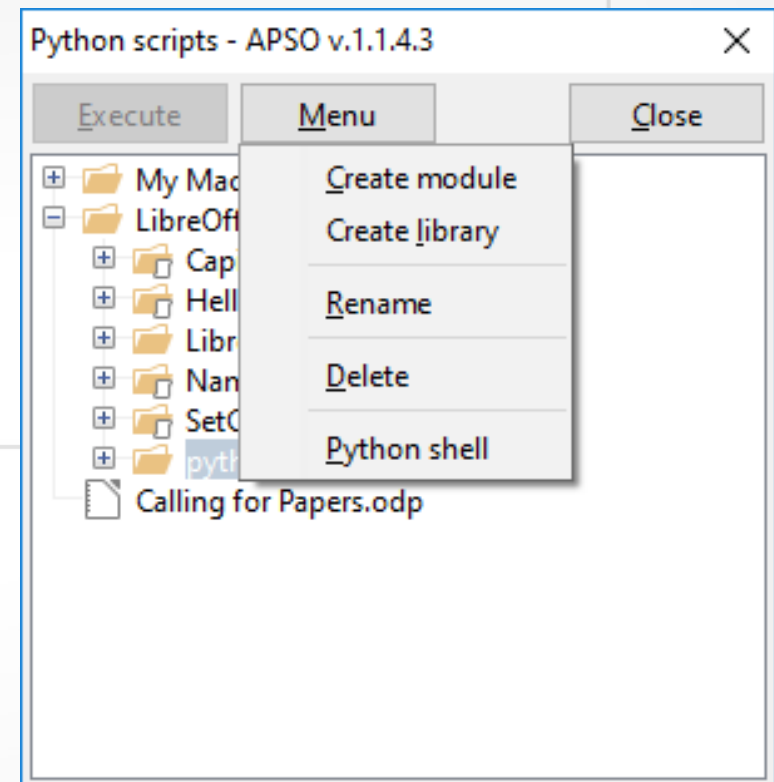
Read Evaluate Print Loop
see also Mri, Xray

Altern. Python Script Organizer



More although..

“



“

..Less

LibreOffice Basic

- Enumerations
- Named Arguments
- Variable args list
- Object-oriented Programming
- Custom Error codes
- Opt. Compatible
- Enum
- ParamArray
- Optl arg=default
- Opt. ClassModule
- Err object

Compiler VBA options

Option Compatible

ParamArray

arg = defaultValue

Option

ClassModule

Initialize

Terminate

Property Get/Set

VBA Enum.erations

- Long datatypes
 - Enum statement
- User defined typed
 - Type .. End Type statement

- Collection class

Note: Fun with MsgBox & Print

MsgBox buttons= prompt=, title=

Print default=, prompt=, title=

Tip: Spreadsheet Functions

Don't reinvent the wheel
reuse Calc functions instead !

`com.sun.star.sheet.FunctionAccess`

Basic: Unit Testing

Basic Test-Driven Development using Python unittest standard module in conjunction with API Scripting Framework

Rules of thumb

1. Stick to LibreOffice Basic + API
2. Extend with VBA 'Option Compatible'
3. OOP with 'Option ClassModule'
4. 'Option VBASupport 1' with caution
 - Warning: Affects Basic behaviour
 - Limit it to doc conversion scenarii
5. avoid CompatibilityMode(bool)
 - Bad coding practice

e-References

local/online **Help**

Basic Macros and Scripting

Wiki

Designing & Developing Python Apps



Q & A

Thanks !