[GRASS-user] Using Python scripts that call GRASS modules or access GRASS layers from outside the GRASS.app with Mac OS X: a little summary

Martin Laloux martin.laloux at gmail.com
Sun Jul 8 07:33:27 PDT 2012

- Previous message: [GRASS-user] r.sun in python loop
- Next message: [GRASS-user] Using Python scripts that call GRASS modules or access GRASS layers from outside the GRASS.app with Mac OS X: a little summary
- **Messages sorted by:** [date] [thread] [subject] [author]

```
*1) if you want to use the GRASS modules from outside in the shell
(terminal) without opening the GRASS application
*
GRASS_and_Python <http://grass.osgeo.org/wiki/GRASS_and_Python> gives
examples to set the environment variables to call the GRASS modules from
outside for Windows and Linux, but nothing for Mac OS X. The solution is
easy if you know the structure of the applications of William Kyngesburye:

*export GISBASE="/Applications/GRASS-6.4.app/Contents/MacOS"*
export PATH="$PATH:$GISBASE/bin:$GISBASE/script:$GISBASE/lib"
export PYTHONPATH="${PYTHONPATH}:$GISBASE/etc/python/"
export PYTHONPATH="${PYTHONPATH}:$GISBASE/etc/python/grass"
export PYTHONPATH="${PYTHONPATH}:$GISBASE/etc/python/grass/script"
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$GISBASE/lib"
export GIS_LOCK=$$
export GISRC="/Users/username/.grassrc6"

if you have not previously defined what is in PATH  in .bash_profile
(as advocated
by William Kyngesburye for the frameworks)  you can use the solution given
by ijufuy in GRASS GIS programming with Python on Mac OS
X<http://yfujimoto.blogspot.be/2011/06/grass-gis-programming-with-python-on.html>
,
adapting it to your versions of GDAL or PROJ frameworks
If you want to use the applications of  Michael Barton:

*export GISBASE="/Applications/GRASS/GRASS-x.x.app/Contents/MacOS" *


After, you can call the GRASS modules with Python and grass.script in the
shell without opening  GRASS-6.x.app:

$ python
Python 2.6.1 (r261:67515, Jun 24 2010, 21:47:49)
[GCC 4.2.1 (Apple Inc. build 5646)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> import grass.script as grass
>>> import grass.script.setup as gsetup
>>> gisbase = os.environ['GISBASE']
>>> gisdb="/Users/username/grassdata"
>>> location="geol"
>>> mapset="mymapset"
>>> gsetup.init(gisbase, gisdb, location, mymapset)
>>> # *table*
>>> desc = grass.parse_command('db.describe', flags='c', table="bxltot")
>>> dict.keys(desc)
['Column 10: Z10:INTEGER:11', 'Column 9: epbxl:INTEGER:11', 'Column 14:
BXL_COR:INTEGER:11', 'Column 4: IDENT:INTEGER:11', 'Column 8:
KOR:INTEGER:11', 'Column 1: cat:INTEGER:11', 'Column 6: LED:INTEGER:11',
'ncols: 14', 'nrows: 47', 'Column 5: Z:INTEGER:11', 'Column 13:
Z_BXL:INTEGER:11', 'Column 2: IGN:DOUBLE PRECISION:20', 'Column 12:
KOR10:INTEGER:11', 'Column 3: AFFLEUREME:DOUBLE PRECISION:20', 'Column 11:
BXL10:INTEGER:11', 'Column 7: BXL:INTEGER:11']
>>> # *geometry*
>>> vector = "bxltot"
>>> points = grass.read_command("v.to.db", flags="p", map=vector,
type="point", option="coor", units="meters", quiet="True")
>>> pt = points.split("\n")
>>>> xyz = []
>>> for i in pt:
...          xyz.append(pt[0].split("|"))
>>> xyz
[['1', '114718.535582253', '119568.077575195', '0'], ['1',
'114718.535582253', '119568.077575195', '0'], ...
```

*2) If you only want to have access to the layers of GRASS, use the
GDAL/OGR Python bindings*

```
>>>  from osgeo import ogr
>>> # open grass vector layer
>>> ds =
ogr.Open('/Users/username/grassdata/geol/mymapset/vector/bxltot/head')
>>> # vector layer
>>> layer = ds.GetLayer(0)
>>> layer.GetName()
>>> 'bxltot'
>>> # *table*
>>> ldefn = layer.GetLayerDefn()
>>> schema = []
>>> for n in range(ldefn.GetFieldCount()):
...          fdefn = ldefn.GetFieldDefn(n)
...          schema.append((fdefn.name, fdefn.type))
>>> schema
[('cat', 0), ('IGN', 2), ('AFFLEUREME', 2), ('IDENT', 0), ('Z', 0), ('LED',
0), ('BXL', 0), ('KOR', 0), ('epbxl', 0), ('Z10', 0), ('BXL10', 0),
('KOR10', 0), ('Z_BXL', 0), ('BXL_COR', 0)]
>>> # *geometry*
>>> points = []
>>> for index in xrange(layer.GetFeatureCount()):
...     feature = layer.GetFeature(index)
...     geometry = feature.GetGeometryRef()
...     points.append((geometry.GetX(), geometry.GetY()))
>>> points
[(114718.535582253,119568.077575195), (114718.535582253,119568.077575195),
....
```

3) advantages and disadvantages of the approach

```
If you know Python, you can combine grass.script with all the others
geospatial or scientific modules (and matplotlib, for example, even in the
GRASS Python shell).


[image: Images intégrées 1]
You can also combine grass.script with qgis.core or r2py (R) or serve
locally the GRASS layers with Django/GeoDjango.

The only disadvantages of the method is when I want to insert the results
in GRASS GIS. If I modify or create geometries, the solutions I found were

   - to use GRASS temporary files as in GRASS and the Python geospatial
   modules (Shapely, PySAL,...)
<http://osgeo-org.1560.n6.nabble.com/GRASS-and-the-Python-geospatial-modules-Shapely-
PySAL-td4985075.html>
   - to  create a new shapefile  and insert it into GRASS.


We should not neglect these other Python modules. They can bring more to GRASS
GIS, especially for vector layers.
-------------- section suivante --------------
Une pi?ce jointe HTML a ?t? nettoy?e...
URL: <http://lists.osgeo.org/pipermail/grass-
user/attachments/20120708/365202cb/attachment-0001.html>
-------------- section suivante --------------
Une pi?ce jointe autre que texte a ?t? nettoy?e...
Nom: matplotlib2.jpg
Type: image/jpeg
Taille: 30973 octets
Desc: non disponible
URL: <http://lists.osgeo.org/pipermail/grass-
user/attachments/20120708/365202cb/attachment-0001.jpg>
```

- Previous message: [GRASS-user] r.sun in python loop
- Next message: [GRASS-user] Using Python scripts that call GRASS modules or access GRASS layers from outside the GRASS.app with Mac OS X: a little summary
- **Messages sorted by:** [date] [thread] [subject] [author]

More information about the grass-user mailing list