Testy GRASS modulů

Skupina C

Děkanová, Pudil

Využití testů v GRASS

- Použití Gunittest package
- Běží na stejném serveru
- Kontrola změn
- Automatické spouštění

Tvorba testů

- Vyzkoušení funkce v GRASS gis
- Přepsání funkce do skriptu v prostředí Python
- Přepsání skriptu do Testovacího prostředí

Testované funkce

- V.random
- R.buffer
- R.stats
- R.region
- R.surf.random
- R.resample

V.random

- Generuje náhodné body
- Nutný definovaný výpočetní region
- Lze vytvořit 3D body

```python
#!/usr/bin/env python3

from grass.gunittest.case import TestCase
from grass.gunittest.main import test

output = 'test01'
output2 = 'test02'
npoints = 100
state = 'boundary_state'
zmin=10
zmax=120

class Test_v_random(TestCase):

    @classmethod
    def setUpClass(cls):
        cls.use_temp_region()
        cls.runModule('g.region', vector=state)

    @classmethod
    def tearDownClass(cls):
        cls.del_temp_region()


    #Checking if number of points equals 100
    def test_num_points(self):
        self.assertModule('v.random', output=output, npoints=npoints,
                          overwrite=True)

        topology = dict(points=npoints)
        self.assertVectorFitsTopoInfo(vector=output, reference=topology)


    #Checking if the map is 3D and number of points is 100
    def test_num_points_3D(self):
        self.assertModule('v.random', output=output, npoints=npoints,
                          zmin=zmin, zmax=zmax,
                          overwrite=True, flags='z')

        topology = dict(points=npoints, map3d=1)
        self.assertVectorFitsTopoInfo(vector=output, reference=topology)

    #Checking if all points are in the polygon boundary state
    def test_restrict(self):
        self.assertModule('v.random', output=output, npoints=npoints,
                          restrict=state, overwrite=True)
        self.assertModule('v.clip', input=output, clip=state,
                          overwrite=True, output=output2)

        self.assertVectorInfoEqualsVectorInfo(output,output2,precision=0.01)


if __name__ == '__main__':
    test()
```

R.surf.random

- Generuje náhodné pixely
- Nutný výpočetní region

```python
#!/usr/bin/env python3

from grass.gunittest.case import TestCase
from grass.gunittest.main import test

output = 'test01'
n=100
s=-100
w=-100
e=100
res=1
min=10
max=40

class Test_v_surf_random(TestCase):

# setting up computational region
    @classmethod
    def setUpClass(cls):
        cls.use_temp_region()
        cls.runModule('g.region', n=n, s=s, e=e, w=w, res=res)

    @classmethod
    def tearDownClass(cls):
        cls.del_temp_region()

# testing if output parameters are between input parameters
    def test_min_max(self):
         self.assertModule('r.surf.random', output=output, min=min, max=max, overwrite=True)
         self.assertRasterMinMax(map=output, refmin=min, refmax=max)

if __name__ == '__main__':
    test()
```

R.resample

- Převzorkování
- Použití metody nejbližšího souseda
- Nutno definovat výpočetní region
- Nutno nastavit rozlišení pixelů a region
- Dnes již nepoužívaný

```python
#!/usr/bin/env python3

from grass.gunittest.case import TestCase
from grass.gunittest.main import test

output = 'test01'
input = 'basins'
resolution = 20
north = 228500
south = 215000
east = 645000
west = 630000
min=2
max=30


class Test_r_resample(TestCase):

    @classmethod

    def setUpClass(cls):
        cls.use_temp_region()

        # change of resolution from 10 to 20
        cls.runModule('g.region', raster=input, res=resolution)

    @classmethod
    def tearDownClass(cls):
        cls.del_temp_region()

    def test_check_resample(self):
        # running r.resample
        self.assertModule('r.resample', input=input, output=output, overwrite=True)

        # testing resolution
        ref_res="nsres=20\newres=20"
        self.assertRasterFitsInfo(raster=output, reference=ref_res)

        # testing coordinates
        ref_extent="north=228500\nsouth=215000\neast=645000\nwest=630000"
        self.assertRasterFitsInfo(raster=output, reference=ref_extent)

        # testing data values
        ref_values="min=2\nmax=30"
        self.assertRasterFitsInfo(raster=output, reference=ref_values)

if __name__ == '__main__':
    test()
```

R.buffer

- Vytváří rastrovou zónu
- Nutné definovat vzdálenost
- Nutné definovat výpočetní region

```python
#!/usr/bin/env python3
from grass.gunittest.case import TestCase
from grass.gunittest.main import test
from grass.gunittest.gmodules import call_module
import grass.script as gscript

output = 'test05'
buffer = 'basins'
distance1=100
distance2=200
ref_stats = '1 1554947\n2 58014\n* 412039'

class Test_r_buffer(TestCase):
    @classmethod
    def setUpClass(cls):
        cls.use_temp_region()
#         cls.runModule('g.region', raster=buffer)


    @classmethod
    def tearDownClass(cls):
        cls.del_temp_region()

#create buffer with distance 100 metres
    def test_check_distance(self):
        self.assertModule('r.buffer', output=output, input=buffer, overwrite=True, distances=distance1)
        # self.assertRastersDifference(actual='buffer', reference='output', precision=0.5, statistics=None)

        stats = call_module('r.stats', input=output, flags='c').rstrip('\n')
        self.assertEqual(stats, ref_stats)

if __name__ == '__main__':
    test()
```

Nahrání do GRASS

- Vytvoření PULL REQUESTu
- https://github.com/OSGeo/grass/pull/704

Děkujeme za pozornost

- Děkanová Lucie, Josef Pudil