Package Installation for Geodata Processing 2025

Installiere zuerst über ein Terminal:

```
conda create --name gpETHZ2025 python=3.12
conda activate gpETHZ2025
conda install -c conda-forge notebook
conda install -c conda-forge gdal
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
import sys,os

!pip install gdal
!pip install psycopg2-binary
!pip install shapely
!pip install fiona
!pip install rasterio
!pip install geopandas
!pip install folium
!pip install ipython-notebook
!pip install leafmap
```

If the following commands may not be installed via Jupyter Notebook. Try it and if it fails, run the following commands in a command shell:

```
import os
# if there is a problem with ipyleaflet run these two commands
# if they fail in jupyter run them on the command-line-interface
'''
os.system('conda install -c conda-forge ipyleaflet')
os.system('conda install -c conda-forge ipython-notebook')
'''
```

Test the Installation

```
import importlib
def importPackages(modules):
    for library in modules:
        try:
            if library == 'gdal':
                exec("from osgeo import gdal")
                print("Package {} successfully imported!".format(library))
            else:
                exec("from {module} import *".format(module=library))
            print("Package {} successfully imported!".format(library))
        except Exception as e:
            print("\n\n E R R O R !!! With Package {}".format(library))
            print(e)
    #print(dir()) # Exactly working as thought
pckList = []
pckList.append('gdal')
pckList.append('psycopg2')
pckList.append('shapely')
pckList.append('fiona')
pckList.append('rasterio')
pckList.append('geopandas')
pckList.append('folium')
pckList.append('json')
pckList.append('os')
pckList.append('sys')
pckList.append('leafmap')

importPackages(pckList)
```

Test Database connection to PostgreSQL/PostGIS

```
import psycopg2
def postgres_test(dbn,h,u,pwd,p):
    try:
        conn = psycopg2.connect("dbname={} host={} user={} password={} port={}".format(dbn,h,u,pwd,p))
        conn.close()
        return True
    except:
        return False

database = "casdb2023"
host = "ikgpgis.ethz.ch"
usr = "casuser"
pwd = "mrPope-2323-IKG"
port = "5432"
print(postgres_test(database,host,usr,pwd,port))
```

Test Reading Data

Copy from one format (Esri Shape) to another (MapInfo) using commandline tool `ogr2ogr`

```
import os
os.system('ogr2ogr -f "MapInfo File" ../Data/Gemeinden_Solothurn.tab ../Data/Gemeinden_Solothurn.shp')
os.system('ogr2ogr -f "GeoJSON"  -t_srs "EPSG:4326" ../Data/Gemeinden_SolothurnWGS84.json ../Data/Gemeinden_Solothurn.shp')
```

Read some data..

You should see something like:
`<osgeo.ogr.DataSource; proxy of <Swig Object of type 'OGRDataSourceShadow *' at 0x7fab289df960> > Gemeinden_Solothurn 109 Der Dater`

```
from osgeo import ogr
drv = ogr.GetDriverByName("ESRI Shapefile")
path2ds = "../Data/Gemeinden_Solothurn.shp"
datasource = drv.Open(path2ds)
print(datasource)

mylayer = datasource.GetLayer(0)
print(mylayer.GetName())

ftrcnt = mylayer.GetFeatureCount()
print(ftrcnt)

print("Der Datensatz '%s' hat %i Gemeinden" %(mylayer.GetName(),int(ftrcnt)))
```

Umprojektion

You should see something like:
`Ausgangskoordinaten: POINT (1120351.57 741921.42) Transformierte Koordinaten: POINT (47.3488013802885 -122.598135130878)`

```
from osgeo import ogr
from osgeo import osr
source = osr.SpatialReference()
source.ImportFromEPSG(2927)

target = osr.SpatialReference()
target.ImportFromEPSG(4326)

transform = osr.CoordinateTransformation(source, target)

origpnt = "POINT (1120351.57 741921.42)"
transfpoint = ogr.CreateGeometryFromWkt(origpnt)
transfpoint.Transform(transform)
print("Ausgangskoordinaten: %s" %origpnt)
print("Transformierte Koordinaten: %s" %transfpoint.ExportToWkt())
```

Rasterdaten

You should see something like: `Anzahl Spalten: 5800 Anzahl Zeilen: 4800 Anzahl Baender: 3`

```
from osgeo import gdal

fn = '../Data/ortho14_5m_rgb_solothurn.tif'
ds = gdal.Open(fn)
if ds is None:
    print ('Datensatz %s konnte nicht geöffnet werden!' %fn)
    sys.exit()

#Dimension des Rasterbildes
cols = ds.RasterXSize
rows = ds.RasterYSize
bands = ds.RasterCount

print ("Anzahl Spalten: %d" %cols)
print ("Anzahl Zeilen: %d" %rows)
print ("Anzahl Baender: %d" %bands)
```

Commandline commands

You should see something like:

```
command to run: gdaldem slope ../Data/Elevation_raster.tif ../Data/Ele_slope.tif -s 10000 0...10...20...30...40...50...60...70...80
```

```
imagefilename = "../Data/Elevation_raster.tif"
path = "../Data/"

sloapcommand = 'gdaldem slope %s %sEle_slope.tif -s 10000' %(imagefilename,path)
print ("command to run: %s" %sloapcommand)   #gdaldem slope input_dem output_slope_map

os.system(sloapcommand)
```

Geowebservices

You should see something like:

```
Successfully downloaded resource http://wms.geo.admin.ch/?SERVICE=WMS&REQUEST=GetMap&VERSION=1.3.0&LAYERS=ch.bfs.arealstatistik-198
```

```
# -*- coding: utf-8 -*-
import os, shutil, sys
import urllib.request
from osgeo import gdal
from osgeo.gdalconst import *

def download(url, dest, fileName=None):
    try:
        r= urllib.request.urlopen(url)
        fileName = os.path.join(dest, fileName)
        with open(fileName, 'wb') as f:
            shutil.copyfileobj(r,f)
        r.close()
        print("Successfully downloaded resource {}".format(url))
    except:
        print("ERROR Downloading resource {}".format(url))

path2save2 = "Data/" #Zielpfad
wmsfile = "wms.gif"
wmslink = "https://wms.geo.admin.ch/?SERVICE=WMS&REQUEST=GetMap&VERSION=1.3.0&LAYERS=ch.bafu.bundesinventare-bln&STYLES=default&CI
download(wmslink,path2save2,wmsfile)
```

Shapely

You should see something like:

```
Gemeinde Rohr hat folgenden Zentroid: (638811.067724, 251323.787658) und folgende Flaeche 2229578.988870m2
```

```
from osgeo import ogr
import shapely.wkt

shapefile = ogr.Open("../Data/Gemeinden_Solothurn.shp")
if shapefile is None:
    print ("Datensatz konnte nicht geoeffnet werden.\n")
    sys.exit()

layer = shapefile.GetLayer(0)

#Gemeindegeometry extrahieren:
geometry = None
cnt = 0
for feature in layer:
    while cnt < 1:
        #Extract Gemeinde-Name
        gemname = feature.GetField("gmde_name")
        #Get Geometry (Polygon)
        gemgeometry = feature.GetGeometryRef()
        #"Convert" Geometry to shapely-geometry
        gemgeomaswkt = gemgeometry.ExportToWkt()
        shapelypolygon = shapely.wkt.loads(gemgeomaswkt)
        #Extract Centroid
        centroid_point = shapelypolygon.centroid
        x=centroid_point.x
        y=centroid_point.y
        area = shapelypolygon.area
        #Printout Information
        print ("Gemeinde %s hat folgenden Zentroid: (%f, %f) und folgende Flaeche %fm2" %(gemname, x, y, area))
        cnt += 1
```

Fiona

You should see something like:

```
Anzahl Datensätze: 109 Format: ESRI Shapefile Geo-Referenzsystem: {'proj': 'somerc', 'lat_0': 46.9524055555556, 'lon_0': 7.43958333
```

```
import fiona

c = fiona.open('../Data/Gemeinden_Solothurn.shp', 'r')
print("Anzahl Datensätze: %i " %len(list(c)))
print("Format: %s" %c.driver)
print("Geo-Referenzsystem: %s" %c.crs)
print("Ausdehnung: %s" %str(c.bounds))
```

Folium

An interactive map located at EHT ZH Hönggerberg with a pin should appear.

```
import folium, json
m = folium.Map(location=[47.40875, 8.50778], zoom_start=17)

folium.Marker(
    location=[47.40875, 8.50778],
    popup="ETH",
    icon=folium.Icon(color="red", icon="info-sign"),
).add_to(m)


rfile = open('../Data/Gemeinden_SolothurnWGS84.json', 'r', encoding='utf-8').read()
jsonData = json.loads(rfile)
style_function = {
                  'fillColor': 'white',
                 }
folium.GeoJson(jsonData, name='json_data'#,
               #style_function=lambda x: style_function
              ).add_to(m)


m
```

```
import folium
m = folium.Map(location=[47.27075, 7.70023], zoom_start=10)

# Reading Polygondata - municipalities
rfile = open('../Data/Gemeinden_SolothurnWGS84.json', 'r', encoding='utf-8').read()
jsonData = json.loads(rfile)        # Reading as dictionary
style_function = {
                  'fillColor': 'white',
                 }
folium.GeoJson(jsonData, name='json_data',
               style_function=lambda x: style_function        # style_function has to be a function which calls dictionary
              ).add_to(m)        # Overlay on map


m.save('../Data/gemSoFolium.html')
m
```

Test Leafmap

```
import leafmap.foliumap as leafmap
m = leafmap.Map(center=(47.5, 7.65), zoom=12)
m.add_basemap("OpenTopoMap")

'''
naip_url = "https://www.mrlc.gov/geoserver/mrlc_display/NLCD_2019_Land_Cover_L48/wms?"
m.add_wms_layer(
    url=naip_url,
    layers="NLCD_2019_Land_Cover_L48",
    name="NLCD 2019",
    attribution="MRLC",
    format="image/png",
    shown=True,
)
'''
pk25 = "https://wms.geo.admin.ch/?"
m.add_wms_layer(
    url=pk25,
    layers="ch.swisstopo.pixelkarte-farbe-pk25.noscale",
    name="PK25",
    #attribution="MRLC",
    format="image/png",
    shown=True,
)

#m.add_legend(title="Pixelkarte 1:25'000")
m
```