Universidad Autónoma del Estado de México
Facultad de Geografía

Programación con Python

Orientado a la automatización de procesos en QGIS

Ing. Adonai Emmanuel Nicanor Bautista

Introducción a la automatización con Ráster

- Gracias a la API de QGIS, podemos tener acceso a las distintas librerías y funciones propias del SIG permitiendo explotar estas a su máximo potencial a través de Python.
- Las funciones que serán aplicadas en este ejercicio son:
  - Lectura de imágenes ráster
  - Suma y comparación de capas ráster
  - Almacenamiento de capas ráster

Introducción a la automatización con Ráster

La primer librería a usar es **QgsRasterLayer** la cual permite abrir un archivo vectorial a través de su **ruta de ubicación**.

- **QgsRasterLayer** recibe 3 parámetros

  1) La ruta (path) del archivo.

  2) El nombre designado al archivo.

  3) El parámetro “ogr” para leer archivo

```
from qgis.core import *
```

- **Nota:** Se debe realizar la importación de la librería

Introducción a la automatización con Ráster

- Una vez que se leyó el archivo, es necesario crear una instancia de éste para mostrarlo en el workspace y tener acceso a todos su atributos, tanto al objeto mismo como a los propios del vectorial.
- **QgsProject.instance().addMapLayer(capa)** permite crear está instancia y facilita tener acceso a la capa ráster en cualquier momento.
- **QgsProject.instance()** por si mismo deja tener acceso a distintas funciones como:
  - Agregar las capas - **addMapLayer(capa)**
  - Leer capa por nombre – **mapLayersByName(“nombre”) -** Lista

Introducción a la automatización con Ráster

- **QgsProject.instance()** por si mismo deja tener acceso a distintas funciones como:
  - Leer todas las capas que se encuentren en el workspace **(habilitadas) – QgsProject.instance().mapLayers() -** Es un dict
  - QgsProject.instance().mapLayers()**.values()**
- **Es importante recordar que todos son objetos ráster.**
  - De las métodos que más se usarán serán:
    - **source() -** path del archivo
    - **name() -** nombre de la capa

Introducción a la automatización con Ráster

- **Es importante recordar que todos son objetos ráster.**
  - De las métodos que más se usarán serán:
    - **width()** - Ancho de la imagen ráster (columnas)
    - **height()** - Largo de la imagen ráster (renglones)

Introducción a la automatización con Ráster

- **Librería Numpy**: Especializada en análisis de datos y calculo numérico de grandes volúmenes de datos. Crea objetos (matrices) multidimensionales. Incluye métodos básicos para trabajar y manipular a estas.
- De los métodos más utilizados son:

```
import numpy as np
```

  - Obtener valor mínimo – obj_numpy**.min()**
  - Obtener el valor máximo – obj_numpy**.max()**
  - Reemplazar valores en la matriz – np**.place(obj_numpy, condición, [valor])**
  - Copiar un objeto numpy – obj_numpy**.copy()**
  - Identificar valores únicos de un objeto numpy – **np.unique(obj_numpy)**

Introducción a la automatización con Ráster

- **Librería Numpy**: Especializada en análisis de datos y calculo numérico de grandes volúmenes de datos. Crea objetos (matrices) multidimensionales. Incluye métodos básicos para trabajar y manipular a estas.
- De los métodos más utilizados son:
  - Obtener número de renglones y columnas – obj_numpy**.shape()** - Tupla

Introducción a la automatización con Ráster

- Para leer una capa ráster como un objeto numpy, se necesita importar a GDAL (Geospatial Data Abstraction Library)

```
from osgeo import gdal
```

- Obj = gdal.**Open(Path)** – Recibe la ubicación de la capa ráster (Path)
- Obj.**ReadAsArray()** - Devuelve la matriz que componen al ráster como un objeto numpy, a este objeto se pueden aplicar los métodos vistos con anterioridad.

Introducción a la automatización con Ráster

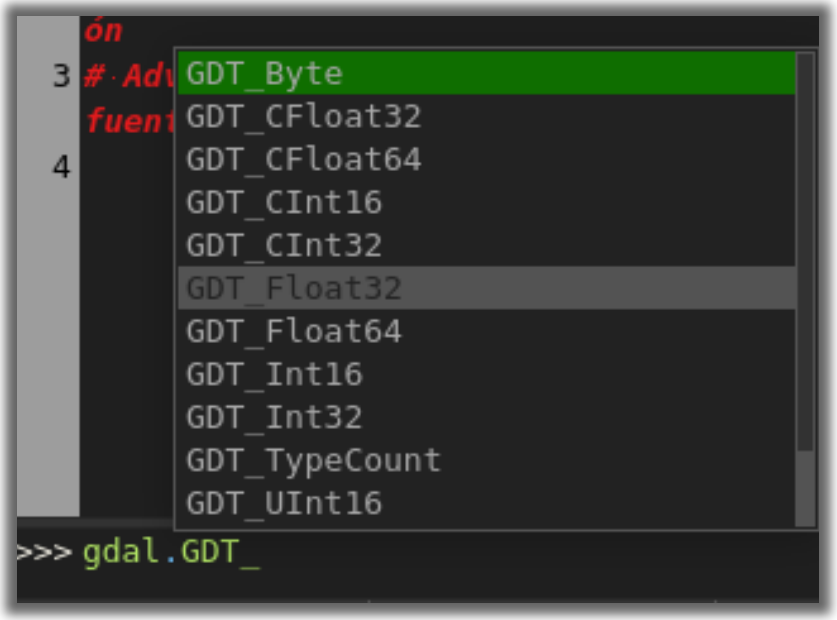
- Para extraer las propiedades de la capa como: La transformación y su proyección.
  - Obj.**GetGeoTransform()** - Retorna una tupla con los valores de la transformación de la imagen
  - Obj.**GetProjection()** - Retorna un string con los valores de la proyección de la imagen.

Introducción a la automatización con Ráster

- Para almacenar capas ráster desde un objeto numpy:
  - Se requiere de una capa auxiliar, la cual cederá algunas de sus propiedades
    - El largo, el ancho, la proyección y la transformación de la imagen.
  - **Modelo = gdal.Open(path)** – Extraer las propiedades.
  - **Driver = gdal.GetDriverByName(“GTiff”)** – Estableciendo que será un ráster para almacenar.
  - **aux = modelo.ReadAsArray()** - Largo y ancho

Introducción a la automatización con Ráster

- Para almacenar capas ráster:
  - **new_raster = driver.Create(path_extension, ancho_X, largo_Y, gdal.GDT_TIPO)**



Introducción a la automatización con Ráster

- Para almacenar capas ráster:
  - **new_raster.SetGeoTransform(modelo.GetGeoTransform()) -** Configurando la transformación de la imagen auxiliar.
  - **new_raster.SetProjection(modelo.GetProjection()) -** Configurando la proyección de la imagen auxiliar.
  - **new_raster.GetRasterBand(1).WriteArray(obj_numpy) –** Guardando el objeto numpy en el nuevo ráster.
  - **Del new_raster –** Eliminando de memoria la creación del objeto.