guia_gis_open

July 11, 2018

1 Guía tecnologías GIS open

1.1 Python

Lenguaje interpretado multipropósito y multiplataforma en el que se intentan seguir los siguientes principios, el también llamado *Zen de Python*, que busca un fácil aprendizaje y hacerlo los más amigable posible.

```
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Para introducirse en él hay muchísimos recursos de calidad pero si el inglés no es problema recomiendo seguir la documentación oficial, como por ejemplo esta guía inicial.

Como lenguaje interpretado que es siempre habrá que lanzar el interprete previamente y pasar como argumento el fichero *.py* que se quiera ejecutar. Si no se pasa ningún fichero entonces se arrancará sesión interactiva en la que podremos probar cualquier sentencia.

```
python hello_world.py
```

Para añadir funcionalidad se usan librerías o paquetes que se gestionan a través de PyPi:

```
pip install geopandas
```

El intérprete guardará los paquetes instalados en paths relativos al directorio de instalación donde reside. A partir de esos paths más el directorio de trabajo donde se encuentra el intérprete se creará el espacio de nombres que es desde donde se podrán importar paquetes y módulos.

```
import lector_csv
lector_csv.leer_csv('usuarios.csv')
```

1.2 Conda

Gestor de paquetes y entornos para poder tener distintos intérpretes `Python` con distintas configuraciones en una misma máquina. En esta guía inicial se explica cómo realizar la instalación y gestionar entornos a través de la herramienta.

Lo interesante de utilizar `conda` para gestionar `python` es que ya viene con una serie de paquetes instalados como *`jupyter`* y *`pandas`*, además de poder instalar librerías de otros lenguajes, no sólo de `python` (*`oracle_instant_client`*, *`nodejs`*).

Así se instalarán paquetes en el entorno activo:

```
conda install -c conda-forge geopandas
```

1.3 Jupyter

Tecnología que permite crear *notebooks*, documentos interactivos que se pueden mostrar en el navegador web y que permiten combinar scripts de código con distintas visualizaciones como MarkDown, widgets de javascript interactivos, imágenes, vídeos, ... En la documentación se puede encontrar esta explicación más concisa de lo que es un `jupyter notebook`.

`Jupyter` es la evolución de Ipython que se podría definir como una consola interactiva de `python`. De hecho desde los `notebook` se puede acceder a una serie de comandos de `IPython` que se pueden conocer ejecutando `%quickref`

JupyterLab Hace unos meses `Jupyter` evolucionó hacia `JupyterLab` que vendría a ser lo mismo pero con unas serie de mejoras en la interfaz y en las tecnologías web utilizadas que lo hacen más potente y versátil. En su documentación se pueden ver los cambios respecto `Jupyter notebooks`

Análisis de datos - pandas: Sería como la navaja suiza para tratar datos de todo tipo, tanto ficheros planos como pueden ser `csv`, `json`, `excel`,... como datos de distintas `BBDD` tanto `sql` como `nosql` (IOTools). Al intentar trabajar en memoria con los datos cargados en `Dataframes`, es muy rápido y versátil haciendo operaciones que de otro modo serían mucho más lentas y tediosas de realizar. - geopandas: Librería que añade a `pandas` características espaciales apoyándose en otras librerías como `shapely`, que permite operar con las geometrías, y `fiona` para leer y escribir en distintos formatos de ficheros espaciales como `geojson`, `shp` o bases de datos como `Postgis`

1.4 Visualización datos

- Folium: Librería que permite crear cliente `Leaflet`(docs) desde `python` y que permitirá añadirlo a un `jupyter notebook`. En este notebook podemos ver algunas de sus capacidades.

- Ipyleaflet: Librería que permite lo mismo que `Folium` y que quizás está mejor integrada con `JupyterLab`
- PyViz: Conjunto de librerías que buscan facilitar la visualización de datos a través de gráficos y mapas interactivos. En este link podemos ver sus capacidades básicas con mapas y datos espaciales.
- Altair: Otra librería para crear dashboards interactivos que a través de `python` hace uso de la librería de `javascript` Vega.
- Plotly: Librería open pero que permite conectar a servicios freemium en la nube (pago por uso) que utiliza la librería de `javascript` plotly.js que a su vez está montada sobre d3.js y stack.gl.

1.5 Clientes web

- Leaflet: Cliente `javascript` de mapas que trabaja con distintos formatos y en distintas plataformas web y mobile. Además de soportar los ya clásicos formatos `wms` y `wfs`, está muy enfocado a usar los formatos vectoriales `geojson` y `topojson` así como el nuevo formato más optimizado y ligero `vector tiles` (aquí explicación de dicho formato) estructurado en teselas al igual que los formatos `raster`.
- Mapbox: plugin `javascript` que se ha montado sobre `leaflet` con la particularidad de tener una `api` que facilita el acceso a los servicios en la nube que ofrece `Mapbox`. En esta respuesta de `Stackoverflow` se aclaran sus diferencias y similitudes.
- OpenLayers: Otro cliente maduro que ha ido incorporando capacidad para mostrar los nuevos formatos vectoriales que se están estandarizando como `geojson`, `topojson` y `vector tiles`.

1.6 Servidores

- Geoserver: Servidor de distintos formatos espaciales en variedad de webservices y tipos de datos. Su funcionalidad se puede ampliar a través de extensiones. Para estilizar las capas de datos servidas por defecto utiliza el standard de la OGC, el Styled Layer Descriptor o SLD, aunque se pueden utilizar otros también a través de extensiones.