EJERCICIOS CAPTURA Y RECOGIDA DE DATOS

1. In this exercise you will navigate over several example **XML**s and you will create one as an example:
   1. Access to this URL and see how it is structured an XML → [https://www.w3schools.com/xml/plant\_catalog.xml.](https://www.w3schools.com/xml/plant_catalog.xml) 
      1. What contains this catalog? Tipos de plantas
      2. How many properties has the main element of the catalog? 6
   2. Access to this URL to load another XML → <https://www.w3schools.com/xml/note_error.xml>
      1. What happens when you try to load it? No va
   3. Go to <https://codebeautify.org/xmlviewer>and load previous xml url.
      1. Could you identify what is the error and fix it? Una F de sobra cuando lo busco por su URL. Se arregla cambiándolo y dándole a tree view
2. Let’s create an XML file: :
   1. Using this table description create an XML file manually:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | Car Model | Color | Year | Owner Name | Owner Address | Owner Phone |
| 1111CXY | Opel | Red | 2014 | Jacinto Rodriguez | Av. Taronjers 10 | 777777777 |
| 0000DDD | BMW | Blue | 2010 | Alicia García | C/ Colon 9 | 111111111 |

* 1. Go to <https://codebeautify.org/xmlviewer>and load this XML to test that is well formed.
  2. Have you created nested elements? Sí, a mano desde un fichero de texto plano o desde la propia dirección de b. Para abrir algo como <car> tengo que cerrarlo como </car> y así con todas las etiquetas. Si pones varios elementos tienen que tener un nombre común (etiqueta) que los englobe para que funcione (car\_catalog).

<?xml version="1.0" encoding="UTF-8"?>

<car\_catalog>

<car>

<ID>1111CXY</ID>

<Model>Opel</Model>

<color>red</color>

<year>2014</year>

<owner\_name>Jacinto Rodriguez</owner\_name>

<owner\_address>Av Tarongers 10</owner\_address>

<owner\_phone>777777777</owner\_phone>

</car>

<car>

<ID>0000DDD</ID>

<Model>BMW</Model>

<color>blue</color>

<year>2010</year>

<owner\_name>Alicia Garcia</owner\_name>

<owner\_address>C/ Colon 9</owner\_address>

<owner\_phone>111111111</owner\_phone>

</car>

</car\_catalog>

1. **JSON** files management, creating one as an example and accessing to the data:
   1. Take a look to an example Kaggle CSV file of sales <https://www.kaggle.com/kyanyoga/sample-sales-data/version/1#sales_data_sample.csv>
   2. See the content and Download the file.
   3. Go to <https://codebeautify.org/jsonviewer>, and create a JSON of 2 first records of the file. Hay que crear una lista o array de dos elementos al fin y al cabo, porque ha dicho que lo creemos con las dos primeras filas. Un ejemplo para crear JSON es este link: <https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=957:ejemplos-json-archivo-datos-cambio-formato-convertidor-on-line-xml-a-json-y-al-reves-minificar-cu01214f&catid=83&Itemid=212>

{"datos":[{

"order\_number": 10107,

"quantity\_ordered": 30,

"precio": 95.7,

"orderline": 2,

"sales": 2871,

"orderdate": "2/24/2003 0:00",

"status": "Shipped",

"qtr\_id": 1,

"month\_id": 2,

"year\_id": 2002

},

{

"order\_number": 10121,

"quantity\_ordered": 34,

"precio": 81.35,

"orderline": 5,

"sales": 2765.9,

"orderdate": "5/7/2003 0:00",

"status": "Shipped",

"qtr\_id": 2,

"month\_id": 5,

"year\_id": 2003

}

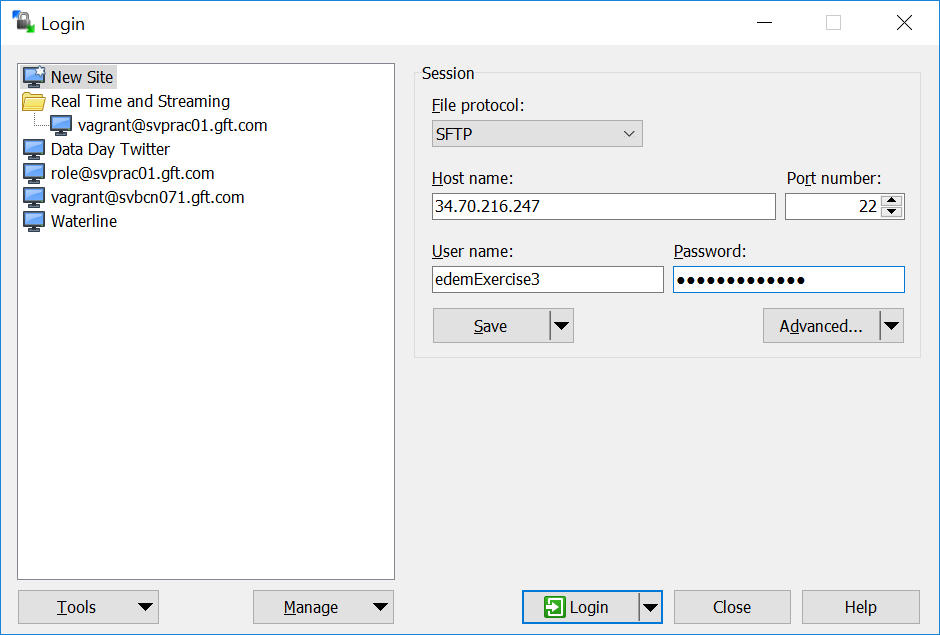
]

}

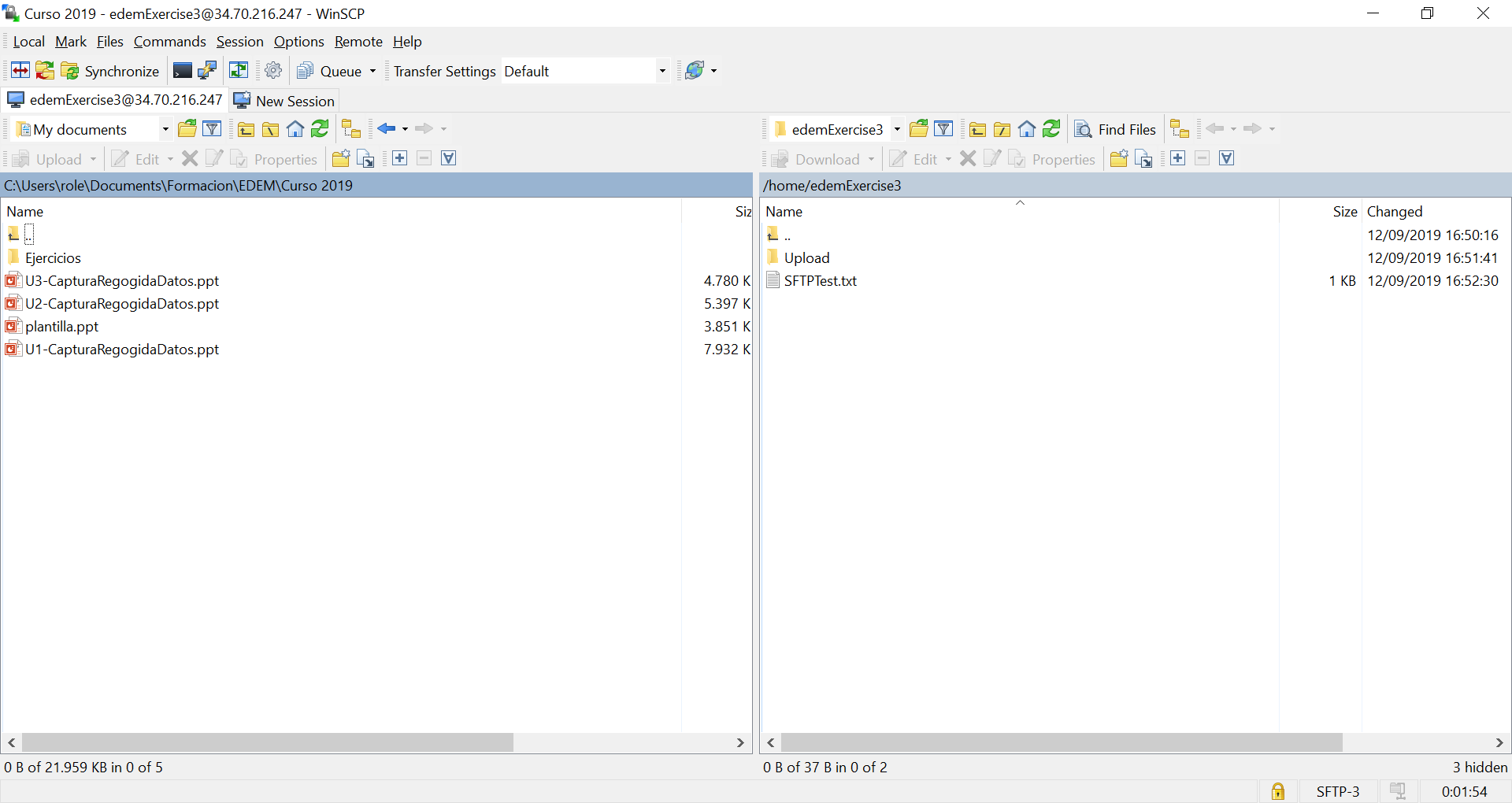
* 1. Now parse it on Python:
     1. Open Jupyther Notebook:
        + Go to git downloaded repo (<https://github.com/rlopezherrero/GFT-EDEM-MasterData/tree/master/CapturaRecogida/docker>)
        + Run ‘docker-compose up -d’ to launch Jupyter Notebook
        + Open <http://localhost:8888>
        + Logon using password ‘edem’.
     2. Load EDEMCRExercise3.ipynb available on downloaded Git repository ([https://github.com/rlopezherrero/GFT-EDEM-MasterData/tree/master/CapturaRecogida](https://github.com/rlopezherrero/GFT-EDEM-MasterData/tree/master/CapturaRecogida/docker))
     3. Follow the instructions: para load o cargar hacerlo con el json minify para que no haya problemas de espacios. Y tener cuidado cuando pone 2, porque Python empieza con 0 y será 1.

1. Let’s play downloading and uploading files using **SFTP protocol**.
   1. If you don’t have it, download and install an SFTP Client: FileZilla
      1. Windows → <https://winscp.net/eng/docs/lang:es>
      2. Linux → <https://filezilla-project.org/download.php?platform=linux64>
      3. Mac Os X → <https://filezilla-project.org/download.php?platform=osx>
   2. On login screen login on remote server
      1. Host → 34.66.154.152
      2. User → edemExercise3
      3. Password → edemExercise3
      4. Port → 22 (Filezilla)

Lo que vamos a hacer es conectarnos a un servidor remoto, cargar un archivo que nos interesa a nuestro local, abrirlo y modificarlo como un archivo normal, y luego volver a subirlo al remoto inicial en un directorio que yo cree ad hoc, donde me diga el ejercicio.



* 1. Now you have started a remote connection and can access to the file system of this machine. You see home directory of edemExercise3 user.



* 1. Download SFTPTest.txt file to your local machine (drag from right and drop on left).
  2. Edit txt file and add a line with a comment of our preference.
  3. Rename the txt file adding your name to the file name (ex. SFTPTestRoberto.txt)
  4. Enter inside Upload folder of the remote machine, and create a directory with your name.
  5. Upload inside this created directory the file you created. Actualizar FileZilla si no aparece modificado el archivo.

1. This exercise you will access to an **API Restful Service** to query (realizar consultas) posts and comments of a Web Service.
   1. Using web browser GET the posts that are available here → <https://jsonplaceholder.typicode.com/posts>
   2. What is the format of the output? Un archivo JSON.
   3. How would you get post 1? Así: https://jsonplaceholder.typicode.com/posts?id=1
   4. This application has also comments, how you will get them?
   5. How you would get all the comments for post 1?
   6. Now let’s do with Python:
      1. Open Jupyter Notebook → <https://jupyter.org/try>
      2. Upload EDEMCRExercise5 available on downloaded git repository ( available on downloaded Git repository ([https://github.com/rlopezherrero/GFT-EDEM-MasterData/tree/master/CapturaRecogida](https://github.com/rlopezherrero/GFT-EDEM-MasterData/tree/master/CapturaRecogida/docker))
      3. Follow the instructions on the Notebook: para acceder a todos de lo que pida es poniendo esto dentro del paréntesis: 'https://jsonplaceholder.typicode.com/posts?userId=1', por ejemplo.

Poner ‘https://jsonplaceholder.typicode.com/comments/1’.

1. On this exercise you will execute a python job to upload a file to a remote server via SFTP. Here you have the api documentation that you can use ( <https://pysftp.readthedocs.io/en/stable/pysftp.html>)
   1. Use jupyther Notebook, also used on exercise 2 ( <https://jupyter.org/try>)
   2. Download template available on GIT ---> <https://github.com/rlopezherrero/GFT-EDEM-MasterData/blob/master/CapturaRecogida/EDEMCRExercise7.ipynb>
   3. Create a local text file with your name (sftpUpload.
   4. Following the template develop a file upload using following connection:
      1. Host → 34.66.154.152
      2. User → edemExercise3
      3. Password → edemExercise3
      4. Folder → Upload/YourName (folder you created on exercise 3)

Esto es para automatizar lo de antes (Ejercicio 4) mediante instrucciones de Python desde Jupyter por ejemplo.

Nota: Si estás trabajando con Jupyter levantado con docker, no mezclar y empezar de repente a trabajar con Jupyter desde Anaconda en local, que se lían.