



Python DBS client from Python 2 to Python 3

Cern Summer Student Projet Report 2021

Author:
Neil FARMER

Kindly supervised by:
Yuyi Guo

Abstract
The aim of this project was to convert a DBS client (written in Python 2) in Python3 and run the unittest.

Background – What is DBS?
"The CMS Data Bookkeeping Service (DBS) provides databases and services to store and access metadata related to the CMS physics event data. […] DBS also contains process-oriented provenance information, including parentage relationships of files and datasets, configurations of processing steps, as well as associations with run numbers and luminosity sections to find any particular set of events within a dataset." [1]

Library used
The DBS client use libcurl and its Python implementation pycurl. To connect to DBS servers, the python script uses a sub-implementation of it, pycurl client. This pycurl client was developed with DBS client. Since pycurl client was also written in Python 2, it was needed to rewrite it in Python 3 too.

Main Changes from Python 2 to Python 3
Python 2 and Python 3 are not radically different. The main changes made in DBS client and pycurl client to convert them in Python 3 were:

- Convert to list the methods applied to the kwargs argument in Python function.
- Be aware of changes between dictionaries/sets and lists.
- Sorting dictionaries
- Change StringIO to ByteIO of elements getting information from pycurl.
- Library

In Python 2, nearly all methods of dictionaries (type of kwargs) used to return list. In Python 3, many of those methods return set. We need to cast those set into list. To do so, we just have to run the method *list().*
As said previously, some methods of list/set/dictionaries may have changed of return type and/or name. All those changes are listed in: The key differences between Python 2.7.x and Python 3.x with examples (sebastianraschka.com).
The *sorted()* function used to sort all types of containers (list, set, dictionary, …) in Python 2. But with Python 3, sets and dictionaries are unordered and can't be sorted or compared. To still work with this function, we now must provide the key argument to sort. This key can be for example a lambda expression.
The information get by pycurl were in Python 2 readable by variable in StringIO. In Python 3, it needed to be a ByteIO variable to get the information. We can then convert the ByteIO into string with the *.decode()* method. The *.decode()* method can receive an argument that provides the type encoding it needs to convert into string. For example, it can be UTF-8.
Finally, some libraries in Python 2 no longer exist in Python 3 or their functions and class have been added natively to Python 3. For example, the library cjson does not have a version for Python 3, but a native json library have been added to Python 3.

Tools
Some tools exist to convert scripts from Python 2 to Python 3, such as the Python library 2to3. But this will mainly remove the import of the library *__future__* and surround every method applied to

dictionaries with *list()* to prevent some *TypeError*. It can be used (and have been used) as a start, but after the conversion has been made, it is important to see all changes apply by comparing the previous script with the new one. Some errors may occur.
After the conversion with the library *2to3,* some search is needed to find all the changes from external libraries, methods name, ….

Pypi

Pypi (Python Package Index) is a repository of software for the Python programming language. In other words, Pypi is where most libraries are stored and updated. Anyone can download any library on Pypi and upload his own library. To get a library from Pypi, we can run the following command:

```
pip install <library name>
```

Before building or uploading your library, download or update the build and twine library:

```
python3 -m pip install --upgrade build
python3 -m pip install --upgrade twine
```

To upload a library on pypi, it is required to create an account. To register go PyPI · The Python Package Index. You will then need to write a setup.py, where you will provide your library name, version, …. Once registered and your setup file written, you will need to build your library distribution. To do so:

```
python setup.py sdist
python setup.py bdist_wheel
```

You are now able to upload your library to pypi:

```
python3 -m twine upload --repository pypi dist/*
```

You can find more detailed tutorials using the following links:
Packaging Python Projects — Python Packaging User Guide
How to upload your python package to PyPi | by joelbarmettlerUZH | Medium

Acknowledgements

I would like to thank my supervisor for her support, for our daily calls to monitor the project progress, and everything that I have learnt during those 9 weeks of CERN summer student project.

References:

[1] Y. G. D. R. M Giffels, «Data Bookkeeping Service 3 - Providing event».