BAM

Bundesanstalt für Materialforschung und -prüfung

Sicherheit in Technik und Chemie

27.11.2025

PYBIS TRAINING – PART I (BEGINNER)

Carlos Madariaga

www.bam.de

Prerequisites / Installation

Overview

Required:

- Python
- Python Package pyBIS

Highly Recommended:

- JupyterLab for interactive use

Recommended:

- MS Visual Studio Code (or other IDE) for development

Python: programming language and environment

Python module/package: package of code for a specific purpose to be used in python (may depend on other packages)

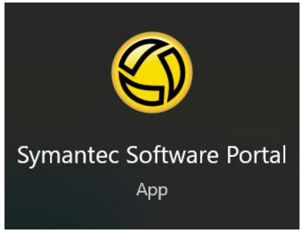
pyBIS: python package for interaction with openBIS

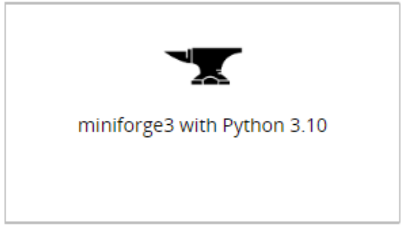
IDE: Integrated Development Environment – code editor and tools for software development

Prerequisites / Installation
Python with Anaconda / Miniforge

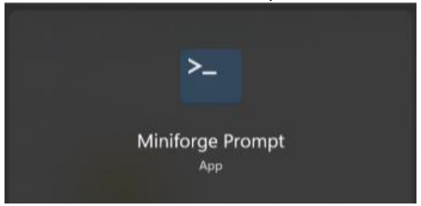
Preferred method:

1. Request installation of *Miniforge3* in the Software Portal





2. Wait for installation to finish
3. Check for “Miniforge Prompt“ (or “Anaconda“ if you had it installed before)



Anaconda: software system to install and manage python and python packages and it's dependencies (not free for BAM)

Miniforge: edition of Conda that only uses free and openly licensed packages from the conda forge project by default free to use at BAM!

Hint: manual Miniconda/Anaconda or pure python installations may also work, but on the BAM client the Software Portal should be used to install python

Prerequisites / Installation
pyBIS package

1. Open *Miniforge Prompt*, a terminal window with a command prompt will appear:
   `(base) PS C:\Users\mmusterm>`

2. Type `pip install pybis` and press ENTER.

3. The pybis package and ist dependencies will be installed. This may run for some seconds or even minutes, depending on your system and internet connection, please be patient

4. After installtion there will be output like `Successfully installed…` and the command prompt.

Hint: if the newest/default version of pyBIS has some errors, try enforce installation of a specific version:

```
pip install pybis=1.36.3
```

Expert Hint: if you already use Anaconda you may create a special environment just for the datastore without changing your existing environment(s):

```
conda create dst python=3.11
conda activate dst
pip install pybis
```

Don't forget to activate this environment when you need it.

Prerequisites / Installation
JupyterLab package

1. Open *Anaconda Powershell Prompt*, a terminal window with a command prompt will appear:
   `(base) PS C:\Users\mmusterm>`

2. Type `pip install jupyterlab` and press ENTER.

3. The jupyter-lab package and its dependencies will be installed. This may run for some minutes, depending on your system and internet connection, please be patient

4. After installtion there will be output like `Successfully installed…` and the command prompt.

JupyterLab: browser-based system to handle code, documentation and results in one document (a.k.a. **Notebook**).

Jupyter: a large software ecosystem that deals with notebooks (Jupyter-Notebooks, Jupyter-Hub, JupyterLab, JupyterLab-Desktop, ...).

Hint: there is a JupyterLab extension for openBIS (*jupyterlab-openbis*). Unfortunately it is not compatible with recent JupyterLab versions.
Pleas do not try to install it!

Python Interpreter

How to start python … and how to get out again…

1. Open *Anaconda Powershell Prompt*, a terminal window with a command prompt will appear:
   ```
   (base) PS C:\Users\mmusterm>
   ```
2. Enter `python`, some version information and a typical python prompt will appear: `>>>`
3. Enter `print(“Hello“)`, the output *Hello* will appear
4. Enter `exit()`, the python system will quit.

Congratulations! You just started python, executed a command and managed to exit from python in a clean way!
As this is not a python tutorial we will not show more of this.
If you need to learn python you will find a lot of tutorials in the web.

You will find a lot of python tutorials in the web. If in doubt use the official source to learn the python basics:

https://docs.python.org/3/tutorial/

PyBIS – the Bare Minimum
Communicate with openBIS in five lines

Open python again (now you know how to do it) and enter the following lines, one after another.

```
from pybis import Openbis
o = Openbis('https://schulung.datastore.bam.de')
o.login('mmusterm', 'bamisgreat')
print(o.get_spaces())
o.logout()
```

The first line loads code from a package (pybis) and makes it available (*Openbis* is now defined and usable). The second line connects to an openBIS instance using the URL and stores the connection in the object *o*. The third line uses this connection and performs a *login* on the server, the connection is now ready to perform actions. In the next line a list of spaces is requested and printed (just an example action). At the end a logout is perfomed.

Please note the ELN analogy: Instead of opening a browser we open python and load pyBIS. Rather than opening a tab and entering the URL we create an Openbis-object with the URL. This object *o* now corresponds to the browser tab – a handle to interact with a specific server. But in both cases we have to login first. After login this tab/object is ready to perform multiple actions – until we logout.

Entering cleartext passwords is not a good idea. Don't worry, we will learn better methods later.

PyBIS – Executing Scripts
Automation

Just copy the five lines of code into a plain text file. You can use something like Windows Editor, Notepad, Notepad++ or similar. Save this file as *pybistest.py* in your *Documents* folder.

Open *Anaconda Powershell Prompt* and run the following command:

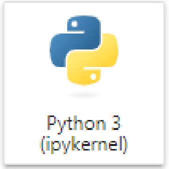
```
python Documents\pybistest.py
```

Now you can edit the code file (e.g. replace *get_spaces* with *get_objects*) and run it again, using the same command.

If you need to develop and run scripts we recommend using an IDE like MS Visual Studio Code, which offer a lot more developer tools than simple text editors.

PyBIS – JupyterLab
Interactive work with notebooks

1. Open *Anaconda Powershell Prompt*
2. Enter *jupyter lab*
3. After some seconds a browser window will appear
4. Create a new Notebook using the following button unter *Notebook*



5. Copy the same five lines from the example above into the cell and press *CTRL+ENTER* (*STRG+ENTER*)

Notebooks - combining advantages of interactive and scripted usage

- Easy editing
- Formatted output (partially)
- Execute in different order
- Add Documentation
- Results may be included
- Client-Server-Architecture
- Collaboration (via plugin)

For a complete introduction to JupyterLab read the section *GetStarted* at **https://jupyterlab.readthedocs.io/**

PyBIS – JupyterLab
More interactive work with notebooks

Some basics before we advance to the example Notebook:

- Splitting code in cells
- Executing in different orders
- Nicer output without *print()*
- Use *Markdown* cells for comments, documentation, and hierarchical structure

Hint: JupyterLab offers more than just Notebooks, there are also command line interfaces, text editors and much more.

Expert Hint: Jupyter is widely known as a Python tool, but a lot of other programming languages can be used within Jupyter.

If you already have code in MATLAB, R, Julia or another language you might try to migrate it to a jupyter notebook.