Tutorial: the CSD Python API installation instructions and notes

The CSD Python API: an introduction

The CSD Python API (Application Programming Interface) enables programmatic access to both Cambridge Structural Database (CSD) data and CSD functionality. The whole spectrum of CSD functionality spanning from CSD-System to more specialist CSD-Materials and CSD-Discovery have been exposed through the CSD Python API. This enables writing of scripts to answer targeted research questions or integrate access to crystal data and CSD functions into user's workflows.

An extensive list of cookbook examples is available from the CSD Python API documentation (https://downloads.ccdc.cam.ac.uk/documentation/API/). They can be modified and tailored to fit your needs and then published into your own *CSD Python API* menu in Mercury and/or Hermes for specialist visualisation and easier communication.

Several mechanisms exist for installing the CSD Python API:

- Using the CSDS installer
- Using standalone CSD Python API installers (both *conda* and *pip* packages)

How to install the CSD Python API

Using the CSD-System installer

The CSDS installer, available on our Download page, https://www.ccdc.cam.ac.uk/support-and-resources/csdsdownloads/ includes a self-contained Python environment called 'miniconda' that has the CSD Python API preinstalled with all of its prerequisites. This will be automatically installed during the CSDS installation process. If the CSDS installer is the mechanism you chose for the CSD Python API then, no additional installation steps are required.

CSDS 2020 Release & Installation Notes
CSDS 2020.1 Windows
CSDS 2020.1 Linux 64-bit
CSDS 2020.1 MacOS
CSD-CrossMiner 2020 User Guide
CSD-CrossMiner 2020.1 Windows
CSD-CrossMiner 2020.1 Linux
CSD-CrossMiner 2020.1 MacOS
CSDS & CrossMiner Download MD5 Validation Checksums
Python 3.7 CSD Python API 3.0.2 Windows 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Windows 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit pip installer
CSD Python API example scripts
CSD Python API utilities

Using standalone CSD Python API installers (both *conda* and *pip* packages are available)

Standalone CSD Python API installers are available for more advanced users who may wish to install the CSD Python API into pre-existing Python environments using either *conda* or *pip* standalone installers.

CSDS 2020 Release & Installation Notes
CSDS 2020.1 Windows
CSDS 2020.1 Linux 64-bit
CSDS 2020.1 MacOS
CSD-CrossMiner 2020 User Guide
CSD-CrossMiner 2020.1 Windows
CSD-CrossMiner 2020.1 Linux
CSD-CrossMiner 2020.1 MacOS
CSDS & CrossMiner Download MD5 Validation Checksums
Python 3.7 CSD Python API 3.0.2 Windows 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Windows 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit pip installer
CSD Python API example scripts
CSD Python API utilities

Conda and *pip* are standard Python package managers that are available on most platforms.

Before installing the CSD Python API, be sure that:

- You have installed and activated the licence for your CSDS package first (see Using the CSD-System installer).
- The CSDS package matches the version of the CSD Python API you wish to install e.g. *CSDS_2020* and *CSD Python API 3.0.0*.

Installing CSD Python API using conda

Anaconda and Miniconda are a free and open-source distribution of Python that aims at simplifying package management and deployment. Anaconda distribution can be downloaded from the Anaconda distribution page here: https://www.anaconda.com/distribution/. It comes with more than 1.500 packages as well as the *conda* package and virtual environment manager.

Alternatively, if you want a minimal installer for *conda* you can install Miniconda instead. Miniconda includes only *conda*, *python*, the packages they depend on, and a small number of other useful packages and it is available here https://docs.conda.io/en/latest/miniconda.html.

It is recommended to install Anaconda if you:

- Are new to *conda* or Python
- Like the convenience of having Python and over 150 scientific packages automatically installed at once
- Have the time and disk space (a few minutes and 3 GB), and/or
- Don't want to install each of the packages you want to use individually.

It is recommended to choose Miniconda instead if you:

- Do not mind installing each of the packages you want to use individually
- Do not have time or disk space to install over 150 packages at once, and/or
- Just want fast access to Python and the *conda* commands and wish to sort out the other programs later.

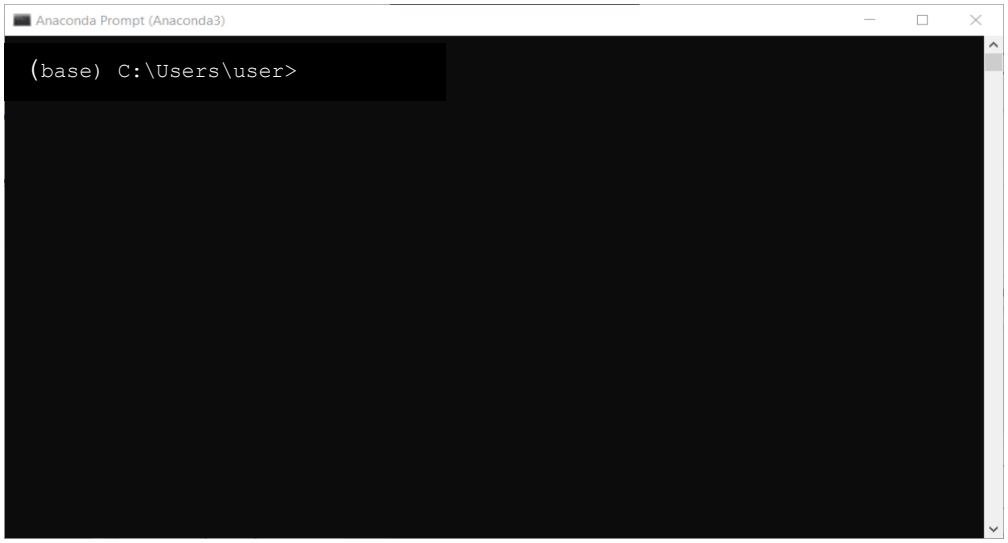
If you want to know more about using Anaconda please check the user guide at the following link https://docs.anaconda.com/anaconda/user-guide/.

1. Download the *Python 3.7 CSD Python API <version> conda package* for your operative system from the download link provided to you via email.

When the download is complete, unzip the *conda* package to a location of your choice. Note that you need to have the CSDS package installed.

2. Be sure that you have a terminal open with the `base` *conda* environment activated. There are two options to start *conda*: you can use a command-line interface or the Anaconda desktop graphical user interface called *Anaconda Navigator*. If you want to know more please check the following link https://docs.anaconda.com/anaconda/user-guide/getting-started/.

   Here we used the Anaconda Prompt or terminal.



Note that, if you have installed a previous version of the CSD Python API (e.g. *CSD Python API 2.0.0*), it is necessary to uninstall it before installing the new CSD Python API.

To do so, use:

```
conda uninstall csd-python-api
```

From here, you have the option to create a new *conda* environment. Note that this is recommended in case you want to experiment with several modules or packages.

You can use the `--clone` argument of the `conda create` command to make an exact copy of the conda `base` environment.

To do so, in your Anaconda Prompt or terminal window type:

```
conda create --clone base -n my_env
```

this will create a new conda environment called `my_env` that is an exact copy of the `base` Anaconda/Miniconda environment.

3. Activate the new `my_env` conda environment using:

```
conda activate my_env
```

To be sure that you are in the desired environment type:

```
conda info --envs
```

a list of environments appears, similar to the following, with the active environment signed with an asterisk (*):

```
conda environments:
#
    base           /home/username/Anaconda3
    my_env    * /home/username/Anaconda3/envs/my_env
```

You can now proceed to install the CSD Python API in the newly created `my_env` conda environment.

4. To install the CSD Python API package in the `my_env` conda environment type:

```
conda install -c <PATH to ccdc_conda_channel> csd-python-api
```

where the `<PATH to ccdc_conda_channel>` is the full absolute path to the unzipped content of the downloaded CSD Python API conda package.

On **Linux and macOS** there are a few more environment variables that must be set for the CSD Python API to communicate with the installed CSD system.

- `CSDHOME` must point to the `CSD_<year>` directory within your CSD-System installation directory. Use the command below to set your `CSDHOME`.

```
$export CSDHOME=/home/my_ccdc_software_dir/CCDC/CSD_<year>
```

- On **Linux** only, assuming that `PYTHONHOME` stores the location of the correct Python installation:

```
$export LD_LIBRARY_PATH=$PYTHONHOME/lib:$PYTHONHOME/lib/python3.7/site-
packages/ccdc/_lib:$LD_LIBRARY_PATH
```

- On **macOS** only:

```
$export DYLD_LIBRARY_PATH=$PYTHONHOME/lib/python3.7/site-packages/ccdc/_lib
$export DYLD_FRAMEWORK_PATH=$PYTHONHOME/lib/python3.7/site-
packages/ccdc/_lib
```

You should now be able to use all the CSD Python API functionality within the `my_env` conda environment.

Installing and using Python and pip package

Most of the recent python packages come with *pip* preinstalled therefore, the installation of the *pip* package should not be required. However, If you do not have already a Python 3.7 installed please download and run the latest installer from https://www.python.org/downloads/ and follow the instructions below.

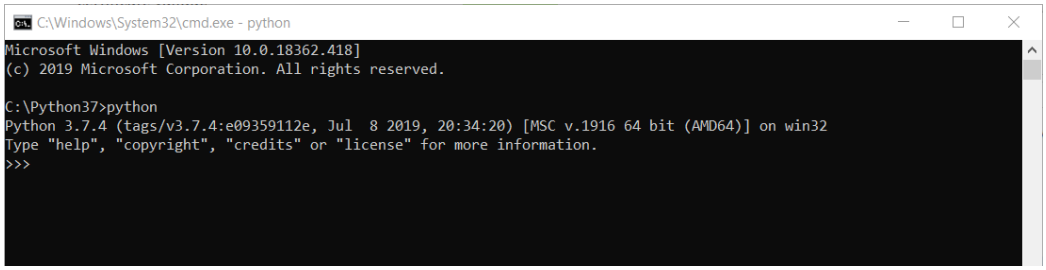
For Windows

1. Download and install the Python 3.7 for windows, by default it will create a `Python37` folder:

```
C:\Python37
```

2. To verify the successful installation of Python3.7, open a *Command Prompt* window by clicking on the Windows **Start** icon and type **cmd**, then click on the *command prompt* icon. From the *Command Prompt* window, navigate to your python installation directory (default is `C:\Python37`).

3. Type `python` from this location to launch the python interpreter, you should have something similar to the following.



   Use the python exit command (`exit()`).

4. To be able to run Python from any location without having to constantly reference the full installation path name, you can add the Python installation path to the Windows `PATH` environment variable .

   Windows allows environment variables to be configured temporarily in a commands prompt shell or permanently at both the User level and the System level.

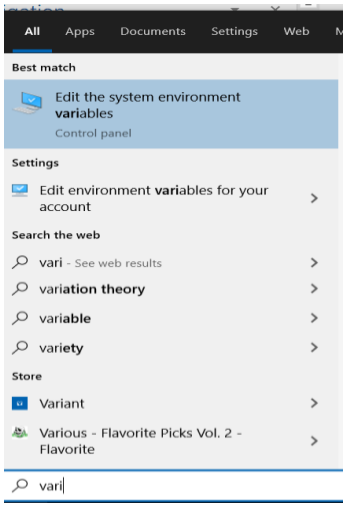
   To temporarily set environment variables, use the Windows *Command Prompt* and the `set` command to include the new installed python directory and python scripts to your `PATH` variable.

```
set PATH=%PATH%;C:\Python37;C:\Python37\Scripts
set PYTHONPATH=%PYTHONPATH%;C:\my_python_lib
```

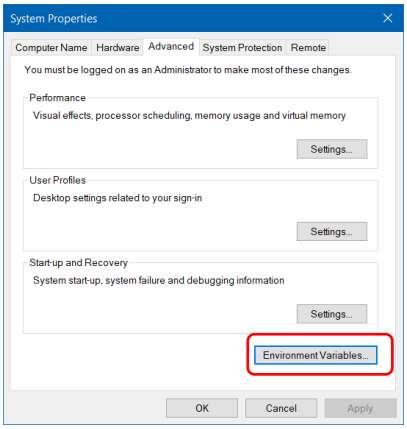
   Including the variable name within percent signs will expand to the existing value, allowing you to add your new value at either the start or the end. These changes will apply to any further commands executed in that console and will be inherited by any applications started from the console.

Note that if you have installed Python on a different location you need to find where your Python is and set that location.

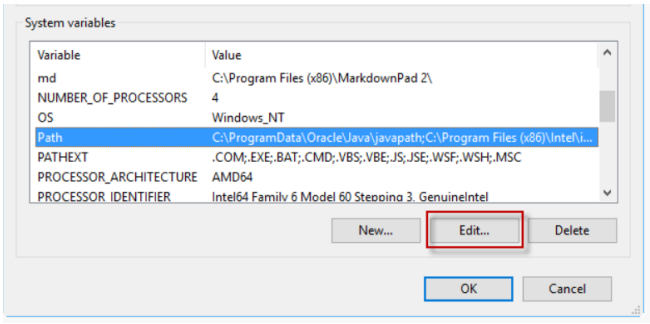
To permanently set environment variables, search for the *System Environment Variables* using the Windows **Start** menu and then click on **Edit the System Environment Variables**.



Click on the **Environment Variables.**



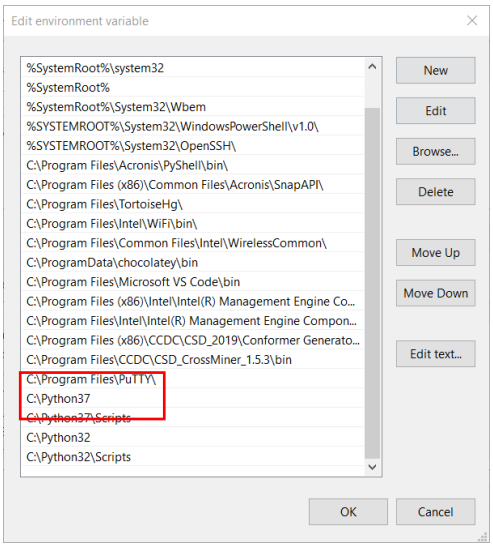
Under the **System variables** section, scroll down and highlight the **Path** variable. Click the **Edit** button.



Add the following routes in order to use the downloaded python and pip.

```
C:\Python37
C:\Pyton37\Scripts
```

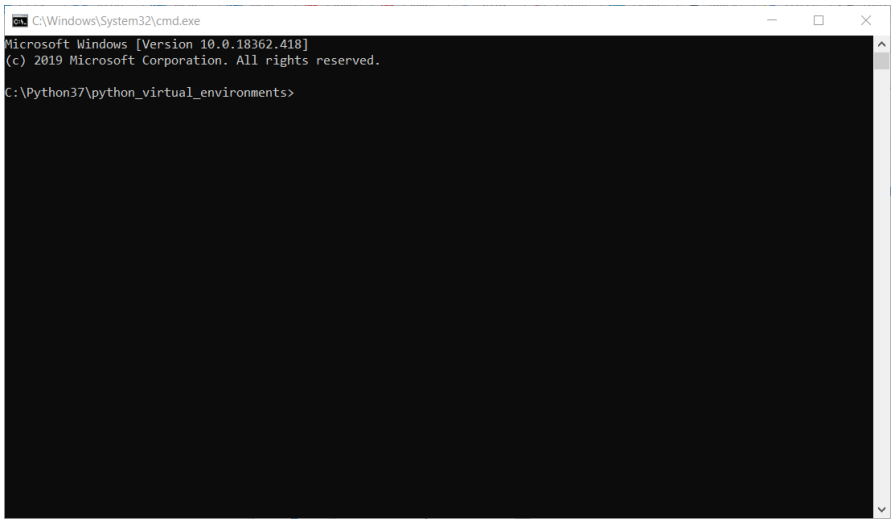
Your **Path** environment should look something like this:



5. As discussed for Anaconda, in most cases, you should use *pip* within a virtual environment only. *Python Virtual Environments* allows you to install different python modules in an isolated location for a specific project, rather than being installed globally. In this way you do not have to worry about affecting other python projects.

   To do so:

   - Create a new folder e.g. *python_virtual_environments* in your `Python37` folder.
   - Open a *Command Prompt* window and navigate to this folder.



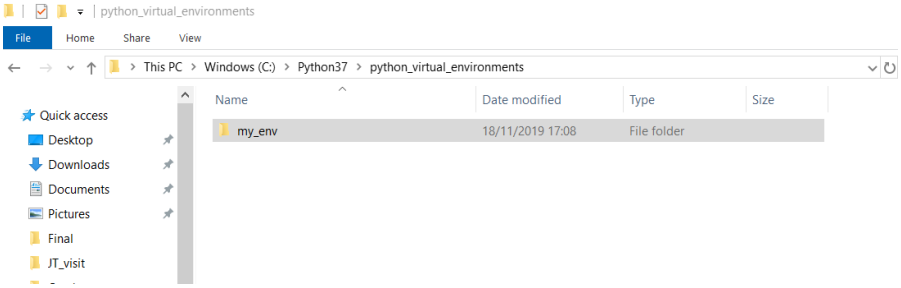
   Then type:

```
<Path_to_the_downloaded_python37>\python -m venv my_env
```

   Alternatively, if you configured the `PATH` variables for your Python installation:

```
python -m venv my_env
```

This command creates a directory called *my_env* that contains all the python executables you need to start working on a separate environment.



Before you can start installing or using packages in your virtual environment, you will need to activate it.

To do so, in your *Command Prompt* window type:

```
.\my_env\Scripts\activate
```

You can confirm you're in the virtual environment by checking the location of your python interpreter, it should point to the `my_env` directory, e.g.

```
where python
.../my_env/bin/python.exe
```

For Linux and macOS

If you do not already have a Python 3.7 installation then please download and run the latest Linux or macOS installer from www.python.org/downloads. To build and install locally use:

```
$ tar zxvf Python-3.7.x.tgz
$ cd Python-3.7.x
$ ./configure --prefix=/home/my/python3.7 --enable-shared
$ make
$ make install
```

Make sure you edit the `.bashrc` environment variables to use the new installed Python.

1. To create a virtual environment, go to your project's directory (or create a new directory e.g. *python_virtual_enviroments)* and run `venv`.

```
python -m venv my_env
```

2. Activate now your environment

```
source my_env/bin/activate
```

You can confirm you're in the virtual environment by checking the location of your python interpreter, it should point to the `my_env` directory e.g.

```
which python
.../my_env/python
```

Installing CSD Python API using pip

Note that, if you have installed a previous version of the CSD Python API (e.g. *CSD Python API 2.0.0*), it is necessary to uninstall it before installing the new CSD Python API.

To do so, use:

```
pip uninstall csd-python-api
```

Download the CSD Python API pip package for your operating system

CSDS 2020 Release & Installation Notes
CSDS 2020.1 Windows
CSDS 2020.1 Linux 64-bit
CSDS 2020.1 MacOS
CSD-CrossMiner 2020 User Guide
CSD-CrossMiner 2020.1 Windows
CSD-CrossMiner 2020.1 Linux
CSD-CrossMiner 2020.1 MacOS
CSDS & CrossMiner Download MD5 Validation Checksums
Python 3.7 CSD Python API 3.0.2 Windows 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Windows 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 Linux 64-bit pip installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit conda installer
Python 3.7 CSD Python API 3.0.2 MacOS 64-bit pip installer
CSD Python API example scripts
CSD Python API utilities

We can now proceed to install the CSD Python API in the new environment (e.g. *`my_env`).*

Please make sure that you have a terminal open with the `my_env` virtual environment activated. You can activate your virtual environment following the instruction in the Installing and using Python and *pip* package.

From the window *Command Prompt* or *terminal* window, you can install the CSD Python API by running one of the following:

On Windows:

```
pip install csd-python-api-|version|-win32-64-py3.7.zip
```

On Linux:

```
pip install csd-python-api-|version|-linux-64-py3.7.zip
```

On macOS:

```
pip install csd-python-api-|version|-mac-64-py3.7.zip
```

If the installation fails to load the required modules with an error similar to ``Unable to load _UtilitiesLib`` then it is likely that your computer does not contain the required Microsoft runtime libraries. These may be installed from: `the Visual C++ Redistributable Package for Visual Studio 2017 https://support.microsoft.com/en-gb/help/2977003/the-latest-supported-visual-c-downloads.

Note that *pip* automatically downloads additional packages such as *`lxml`*. Some company firewalls block this, so you may need to configure an HTTPS proxy. Ask your systems administrator for more information.

On **Linux and macOS** there are a few more environment variables that must be set to the CSD Python API to communicate with the installed CSD system .

- `CSDHOME` must point to the `CSD_<year>` directory within your CSD-System installation directory:

  ```
  $ export CSDHOME=/home/my_ccdc_software_dir/CCDC/CSD_<year>
  ```

- **On Linux only,** assuming that `PYTHONHOME` stores the location of the correct Python installation:

  ```
  $ export
  LD_LIBRARY_PATH=$PYTHONHOME/lib:$PYTHONHOME/lib/python3.7/site-
  packages/ccdc/_lib:$LD_LIBRARY_PATH
  ```

- **On macOS only**

  ```
  $ export DYLD_LIBRARY_PATH=$PYTHONHOME/lib/python3.7/site-
  packages/ccdc/_lib

  $ export DYLD_FRAMEWORK_PATH=$PYTHONHOME/lib/python3.7/site-
  packages/ccdc/_lib
  ```

Note that if you are an associate collaborator you may want to download the additional `ccdc_rp` package from the Download portal https://www.ccdc.cam.ac.uk/support-and-resources/downloads/ e.g. *Python 3.7 CSD Python API 3.0.0 Research Partner Add-on pip package.*

To install it use:

```
pip install csd-python-api-rp|version|.tar.gz
```

You should now be able to use all the CSD Python API functionality within the `my_env` python environment.