Les Houches PNP School- Online tools
Online tools for planetary sciences

Objectives

The objective of this hands-on session is to introduce some of the most common Virtual Observatory (VO) tools, such as `aladin` (Bonnarel et al., 2000) and `TOPCAT` (Taylor, 2005), and a widely used protocol, `cone-search`, to query data from archives.

We will start with a graphical user interface and then move to using `python` scripts to query Application Programming Interfaces (API) and automatize the repetitive tasks.

1   Finding moving objects in an image (field of view)

This exercise introduces the Sky Body Tracker (`SkyBoT`) tool (Berthier et al., 2006). `SkyBoT` is a specialized `cone-search`: it lists all Solar System Objects (SSOs) present within a given field of view **at a given epoch**. Alike all the services hosted at IMCCE on the VO Solar System Portal (VOSSP[1]), `SkyBoT` requests can be submitted in different ways (`aladin`, HTTP, SOAP), and results can be provided in different formats (VOTable, text, html).

We will use `SkyBoT` to illustrate calls to VO services first with a graphical user interface (GUI), `aladin` (Fig. 1), then with a `python` script (in the **XXX** notebook).

Fig. 1: The `aladin` client. Loaded data from the `data exploration` panel are listed as `layers` and displayed in the `sky view`. Coordinates can be entered in the `view control` panel, and many `tools` are available.

---

[1]Forms: https://ssp.imcce.fr/forms
   APIs: https://ssp.imcce.fr/webservices

---

1. Launch `aladin`.

2. Using the `view control`, center de view on (RA,Dec) of (07h 08m, +26d 34m).

3. Open the `Serveur selector`. Keystroke `ctrl-l` or File → `Open server selector`.

There are seven fieds in the pop-up window: `Target`, `Radius`, `Epoch`, `Observer location`, `Search For`, `Max. uncertainty`, and `Display filter`. By default, `aladin` fills the `Target` and `Radius` fields to correspond to current display.

4. Check that the coordinates in `Target` correspond to the value above.

5. Set the `Radius` to 15'.

6. Set the `Epoch` to `2022-06-21T00:00:00`.

7. Let everything else by default and click on `SUBMIT`.

A new layer has appeared, showing the SSOs with their arrow of motion. Like for any other catalog loaded in `aladin`, you can select objects and see their properties in the bottom panel.

8. Select all SSOs. `Ctrl-a` or Edit → `Select all objects`.

9. Click on any of the column in the bottom panel. The histogram of values for all objects will appear at the bottom right.
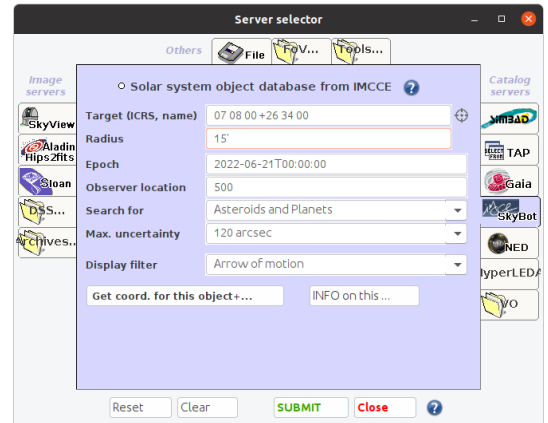
2   In a script

We often process many images. Clicking on a GUI is not the most productive method. We thus often write scripts to perform many automated steps. Most VO services can be called through APIs and be scripted.

Generally, the service has a base `url` to which queries are to be sent. The user choices are submitted through a list of `parameter=value` arguments. Of course, we cannot guess which `parameter` is available, nor what `value` are allowed. So read the documentation!

For instance, we can repeat the query above in `python`:

```python
import requests
import json
import pandas as pd
from astropy.coordinates import Angle

# User choices
ep = '2022-06-21T00:00:00'
ra = '07h08m00'
dec = '+26d34m00'
sr = 15/60
observer = '500'

# Service URL
url = 'http://vo.imcce.fr/webservices/skybot/skybotconesearch_query.php?'

# Query parameters
params = {
    '-ep': ep,
    '-ra': Angle(ra).degree,
    '-dec': Angle(dec).degree,
    '-sr': sr,
    '-mime': 'text',
    '-output': 'all',
    '-loc': observer,
    '-tscale': 'UTC'
    }
```

```
28  # Query the service
29  r = requests.post(url, params=params, timeout=2000)
30
31  # Write results to disc
32  with open("response.txt", "w") as f:
33      f.write(r.text)
34
35  # Load results in a pandas.DataFrame
36  data = pd.read_csv( 'response.txt', sep='|', skiprows=2 )
```

Similarly to the TOPCAT/stilts script in **??**, we first define the parameters (coordinates, radius, epoch) then query the SkyBoT service thanks to the request package. The results are written in a file, and read as a pandas DataFrame. Performing this query for each image acquired by a telescope is very easy (the parameters are contained in the image).

References

Berthier, J., Vachier, F., Thuillot, W., et al. 2006, in Astronomical Society of the Pacific Conference Series, Vol. 351, Astronomical Data Analysis Software and Systems XV, ed. C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, 367

Bonnarel, F., Fernique, P., Bienaymé, O., et al. 2000, A&AS, 143, 33

Taylor, M. B. 2005, in Astronomical Society of the Pacific Conference Series, Vol. 347, Astronomical Data Analysis Software and Systems XIV, ed. P. Shopbell, M. Britton, & R. Ebert, 29