Fig. 1

1. pyVO's new Registry Interface

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

- Status quo ante
- Motivating the change
- The new API

(cf. Fig. 1)

2. pyVO Registry So Far

The legacy pyVO API to the registry has essentially been:

```
registry.search(keywords, servicetype)
  -> service-descriptors
```

– when you wanted infrared spectra, you would look for, roughly, "SSAP services with infrared".

3. Why Touch It?

That's not good enough any more: Services and data collections are *really* not 1:1 any more (extreme case: several 10^4 tables within the single TAPVizieR service).

Also, with libraries like pyVO you don't care too much any more whether you get something through TAP, SCS, or SIAP, as long as you get it.

4. pyVO Registry After

The idea is that you combine Constraints and get back registry records that you can ask about how they can be queried.

To look for resources talking about quasars with redshift columns, you could say:

```
rscs = registry.search(
    registry.Freetext("quasar"),
    registry.UCD("src.redshift"))
```

When the resource you have discovered has SCS and/or TAP endpoints, you can say

```
rscs["III/175"].get_service("conesearch"
  ).search(126, -20), 5)
rscs["III/175"].get_service("tap"
  ).run_sync('SELECT TOP 5 * FROM "III/175"')
```

5. Demo Time

Try it yourself:
http://blog.g-vo.org/media/2022/data-discovery-demo.ipynb

https://blog.g-vo.org/towards-data-discovery-in-pyvo.html tells you how to do this without nuking you system's pyVO.

6. Parting Words

The code discussed here is available starting pyVO release 1.4.

If you run a publishing registry: See how your resources show up in this interface – and if they don't have a tableset yet, *please add one!*

. . . Thanks!