Python iRODS Client v3.1.1

Daniel Moore
Applications Engineer
iRODS Consortium

June 17-20, 2025
iRODS User Group Meeting 2025
Durham, NC

Python iRODS Client Library

Originally developed at CyVerse (then known as the iPlant Collaborative) in 2013 and contributed in 2014 to the newly established iRODS Consortium.

Now an established PyPI package, it is actively kept up to date with any iRODS API changes and is popular as a basis for many other projects.

Thank you to the 51 contributors and many users over the years.

Python iRODS Client from 2.1.0 to 3.1.1 - Major Updates

API endpoints implemented

- GenQuery2 (by @stsnel)
- replica truncate
- library features
- touch (by @korydraughn)
- client_hints (by @stsnel)

Enhancements

- authenticates with iRODS 4.3+ auth framework / plugins
  - native, pam_password
- some iinit-like capability (create authentication files)
- progress bar compatibility (updatables parameter in put/get)

Python iRODS Client Library 3.1.1 - Other Improvements

Further Enhancements

- can now determine server version without first authenticating
- context manager to alter choice of xml parser
- source code reformatted using 'black'
- dropped Python2 support as of v3.0.0

Python iRODS Client Library 3.1.1 - Other Improvements

Bug fixes

- resource redirect is no longer a default
- put and create will conform with force flag constraints
- corrected faulty column mappings
- can configure client using OS environment, without file dependency
- session.clone() now preserves per-connection ticket information
- logging.basicConfig() no longer called from within library
- quieter library exit when errors occur in connection teardown

Python iRODS Client Library 3.1.1 - Examples of Use

To determine server version without first authenticating:

```python
import logging
from irods.helpers import make_session

session = make_session()

if session.server_version_without_auth() > (4,3,4):
    logging.warning("Server may be too recent.")

# Authentication happens on first API exchange with server.
myuser = session.users.get(session.username)

# ...
```

To use xml_mode:

```python
from irods.helpers import XML_Parser_Type, xml_mode, make_session, home_collection

session = make_session()
home_path = home_collection(session)

with xml_mode(XML_Parser_Type.QUASI_XML):
    home_collection = session.collections.get(home_path)
    for d in home_collection.data_objects:
        # Some processing of each data object 'd' goes here!

# Code placed outside the above code block will use the default parser, STANDARD_XML
```

Python iRODS Client Library 3.1.1 - Examples of Use

To use GenQuery2:

```python
import logging
from irods.helpers import make_session, home_collection

session = make_session()
home_path = home_collection(session)

gqo = session.genquery2_object()

enum_cols = lambda colnames_str:dict(
    tuple(reversed(_)) for _ in enumerate(colnames_str.split()))

col_index = enum_cols("""  DATA_NAME
                           COLL_NAME""")

query = f"""SELECT {",".join(col_index.keys())}
            WHERE COLL_NAME = '{home_path}'
            OR (META_DATA_ATTR_NAME = 'moving-to' AND META_DATA_ATTR_VALUE = '{home_path}')"""

result = gqo.execute(query)

for row in result:
    print(f"""Collection Name: {row[col_index['COLL_NAME']]!r}\n"""
          f"""Data Object Name: {row[col_index['DATA_NAME']]!r}\n""")
```

Python iRODS Client Library - Future Work

- Support iRODS 5
- Support pam_interactive authentication plugin
- Integrate Python client testing

Questions?