Project

-------

adodbapi

A Python DB-API 2.0 (PEP-249) module that makes it easy to use Microsoft ADO

for connecting with databases and other data sources using CPython.

Home page: <http://sourceforge.net/projects/adodbapi>

Features:

\* 100% DB-API 2.0 (PEP-249) compliant (including most extensions and recommendations).

\* Includes pyunit testcases that describe how to use the module.

\* Fully implemented in Python. -- runs in current versions of Python 3

\* Licensed under the LGPL license, which means that it can be used freely even in commercial programs subject to certain restrictions.

\* The user can choose between paramstyles: 'qmark' 'named' 'format' 'pyformat' 'dynamic'

\* Supports data retrieval by column name e.g.:

for row in myCurser.execute("select name,age from students"):

print("Student", row.name, "is", row.age, "years old.")

\* Supports user-definable system-to-Python data conversion functions (selected by ADO data type, or by column)

Prerequisites:

\* C Python 3.6 or higher

and pywin32 (Mark Hammond's python for windows extensions.)

Installation:

\* (C-Python on Windows): Install pywin32 ("pip install pywin32") which includes adodbapi.

\* (IronPython on Windows): Download adodbapi from http://sf.net/projects/adodbapi. Unpack the zip.

NOTE: ...........

If you do not like the new default operation of returning Numeric columns as decimal.Decimal,

you can select other options by the user defined conversion feature.

Try:

adodbapi.apibase.variantConversions[adodbapi.ado\_consts.adNumeric] = adodbapi.apibase.cvtString

or:

adodbapi.apibase.variantConversions[adodbapi.ado\_consts.adNumeric] = adodbapi.apibase.cvtFloat

or:

adodbapi.apibase.variantConversions[adodbapi.ado\_consts.adNumeric] = write\_your\_own\_convertion\_function

............

notes for 2.6.2:

The definitive source has been moved to https://github.com/mhammond/pywin32/tree/master/adodbapi.

Remote has proven too hard to configure and test with Pyro4. I am moving it to unsupported status

until I can change to a different connection method.

what's new in version 2.6

A cursor.prepare() method and support for prepared SQL statements.

Lots of refactoring, especially of the Remote and Server modules (still to be treated as Beta code).

The quick start document 'quick\_reference.odt' will export as a nice-looking pdf.

Added paramstyles 'pyformat' and 'dynamic'. If your 'paramstyle' is 'named' you \_must\_ pass a dictionary of

parameters to your .execute() method. If your 'paramstyle' is 'format' 'pyformat' or 'dynamic', you \_may\_

pass a dictionary of parameters -- provided your SQL operation string is formatted correctly.

what's new in version 2.5

Remote module: (works on Linux!) allows a Windows computer to serve ADO databases via PyRO

Server module: PyRO server for ADO. Run using a command like= C:>python -m adodbapi.server

(server has simple connection string macros: is64bit, getuser, sql\_provider, auto\_security)

Brief documentation included. See adodbapi/examples folder adodbapi.rtf

New connection method conn.get\_table\_names() --> list of names of tables in database

Vastly refactored. Data conversion things have been moved to the new adodbapi.apibase module.

Many former module-level attributes are now class attributes. (Should be more thread-safe)

Connection objects are now context managers for transactions and will commit or rollback.

Cursor objects are context managers and will automatically close themselves.

Autocommit can be switched on and off.

Keyword and positional arguments on the connect() method work as documented in PEP 249.

Keyword arguments from the connect call can be formatted into the connection string.

New keyword arguments defined, such as: autocommit, paramstyle, remote\_proxy, remote\_port.

\*\*\* Breaking change: variantConversion lookups are simplified: the following will raise KeyError:

oldconverter=adodbapi.variantConversions[adodbapi.adoStringTypes]

Refactor as: oldconverter=adodbapi.variantConversions[adodbapi.adoStringTypes[0]]

License

-------

LGPL, see http://www.opensource.org/licenses/lgpl-license.php

Documentation

-------------

Look at adodbapi/quick\_reference.md

http://www.python.org/topics/database/DatabaseAPI-2.0.html

read the examples in adodbapi/examples

and look at the test cases in adodbapi/test directory.

Mailing lists

-------------

The adodbapi mailing lists have been deactivated. Submit comments to the

pywin32 mailing lists.

-- the bug tracker on sourceforge.net/projects/adodbapi may be checked, (infrequently).

-- please use: https://github.com/mhammond/pywin32/issues