Generated at Sun Apr 30 02:39:17 2023.

Source at https://jupyterhub-dev.cheme.cmu.edu/user/ccolomb2@andrew.cmu.edu/lab/tree/s23-06682/assignments/project_final/s23project/project_final.ipynb.

Before you turn this problem in, make sure everything runs as expected. First, **restart the kernel** (in the menubar, select Kernel→Restart) and then **run all cells** (in the menubar, select Cell→Run All).

Make sure you fill in any place that says `YOUR CODE HERE` or "YOUR ANSWER HERE", as well as your name and collaborators below:

In [1]:
```
NAME = "Carolina Colombo Tedesco"
COLLABORATORS = "Kareem, Noah, Nicholas, Saaksshi"
```

pip install s23packProject

The final project is to create a small Python package for OpenAlex based on what we have learned so far. You will create the package and host it on GitHUB. You will turn in a pdf of this notebook.

Your tasks are:

1. Create a pip installable Python package in a GitHUB repo that provides an OpenAlex Works class. The class should have methods to get an RIS and a bibtex entry for a DOI. You can reuse code from previous assignments and lectures. Your class should also have a command line utility that prints RIS or bibtex to the terminal.
2. Your package must have some tests that show at least some part of the package works correctly.
3. You should make sure your repo passes black and pylint. Your code should pass both of these.
4. You should setup a GitHUB action that runs your tests
5. You should add an Actions status badge that shows in the README.
6. Your package should also have a license.

Put the URL to your repo in the next cell:

Here is the url for my repo in GitHub:

https://github.com/ccolomb2/s23project

Clone the repo here

You should clone your repo in this folder. Use the tree command to show your repo structure:

```
! tree your-repo-name
```

For my notebook to work, I had to pip install my project right away.

With the cell below, I cloned my GitHub repository. Then I built the package here, to then push back.

In [2]:
```
# ! git clone https://github.com/ccolomb2/s23project.git
```

```
fatal: destination path 's23project' already exists and is not an empty directory.
```

In [3]:
```
!pip install /home/jupyter-ccolomb2@andrew.cm-7277f/s23-06682/assignments/project_final/
```

```
Defaulting to user installation because normal site-packages is not writeable
Processing ./s23project
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: pkg
  Building wheel for pkg (setup.py) ... done
  Created wheel for pkg: filename=pkg-0.0.1-py3-none-any.whl size=3444 sha256=425acc69cc
1576860cb0297ff7223a9ae90ebb5e2a257bd6ed0ec5c28046dc97
  Stored in directory: /tmp/pip-ephem-wheel-cache-03pjajqz/wheels/2a/9a/07/99eda92b5bbf8
9c43718e44e7967436ffa065ea760776dafb8
Successfully built pkg
Installing collected packages: pkg
  Attempting uninstall: pkg
    Found existing installation: pkg 0.0.1
    Uninstalling pkg-0.0.1:
      Successfully uninstalled pkg-0.0.1
  WARNING: The script commandline is installed in '/home/jupyter-ccolomb2@andrew.cm-7277
f/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use
 --no-warn-script-location.
Successfully installed pkg-0.0.1
```

This is the structure of my repo, where one can see the folders and files contained in it.

In [4]:
```
! tree s23project
```

```
s23project
├── build
│   ├── bdist.linux-x86_64
│   └── lib
│       └── pkg
│           ├── __init__.py
│           ├── terminal_command.py
│           ├── test.py
│           └── worksproject.py
├── pkg
│   ├── __init__.py
│   ├── LICENSE
│   ├── README.md
│   ├── terminal_command.py
│   ├── test.py
│   └── worksproject.py
├── pkg.egg-info
│   ├── dependency_links.txt
│   ├── entry_points.txt
│   ├── PKG-INFO
│   ├── SOURCES.txt
│   └── top_level.txt
├── project_final.ipynb
├── __pycache__
│   ├── __init__.cpython-39.pyc
│   ├── test_oa.cpython-39-pytest-7.2.2.pyc
│   └── worksproject.cpython-39.pyc
└── setup.py

7 directories, 20 files
```

Show evidence that your repo passes black and pylint

`black` took care of editing my files with its style. It would only give me a broken heart emoji if something inside my files were not working. Otherwise it gives me the slice of cake. It is unclear to me if it should be interpreted as "piece of cake", "py" as in "pie" and "python" or if it's just a gift for the user. Either way, it is nice.

In [5]:
```
%%bash

black s23project/pkg
```

```
All done! □ □ □
8 files left unchanged.
```

`pylint` at first gave me a lot of things to correct. Mostly, it told me to put docstrings in my functions, remove spaces and rename variables that were out of style or not clear, as the one-letter ones, which are likely considered bad practice. When everything was corrected, I got the score 10/10. Very satisfying, indeed.

In [6]:
```
%%bash

pylint pkg
```

```
--------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 9.84/10, +0.16)
```

Tests

Create one or more tests in the repo that show your package works correctly. Show an example here that your tests work.

Using `pytest`, we can check our class Works returns exactly what is expected from it.

In [16]:
```
! pytest s23project/pkg/test.py
```

```
============================= test session starts ==============================
platform linux -- Python 3.9.7, pytest-7.2.2, pluggy-1.0.0
rootdir: /home/jupyter-ccolomb2@andrew.cm-7277f/s23-06682/assignments/project_final/s23p
roject
plugins: typeguard-2.13.3, anyio-3.6.1
collected 2 items

s23project/pkg/test.py ..                                                [100%]

============================== 2 passed in 0.24s ===============================
```

Make some examples of your package to show it works here

Install the package, and show an example for each method (RIS, and bibtex). Provide some evidence that the examples work correctly and generate valid RIS and bibtex.

We can see below that the package works, and one could paste the url of many papers in the array and get a list of bibtex formatted references, which can be copied and pasted directly in Overleaf.

In [8]:
```
import numpy as np
from pkg import Works

dois = np.array(['https://doi.org/10.1021/acscatal.5b00538', 'https://dx.doi.org/10.1021

for doi in dois:
    w = Works(doi)
    print(w.bibtex())
```

```
@article{J2015,
author = {John R. Kitchin},
journal ={ACS Catalysis},
title = {Examples of Effective Data Sharing in Scientific Publishing},
volume = {5},
issue ={6},
pages = {3894-3899},
year = {2015}
}
@article{J2020,
author = {Joshua Gopeesingh, M. Alexander Ardagh, Manish Shetty, Sean Burke, Paul J. Dau
enhauer and Omar A. Abdelrahman},
journal ={ACS Catalysis},
title = {Resonance-Promoted Formic Acid Oxidation via Dynamic Electrocatalytic Modulatio
n},
volume = {10},
issue ={17},
pages = {9932-9942},
year = {2020}
}
@article{Y2017,
author = {Yixin H. Ye, Ignacio E. Grossmann and José Carlos Pinto},
journal ={Computers & Chemical Engineering},
title = {Mixed-integer nonlinear programming models for optimal design of reliable chemi
cal plants},
volume = {116},
issue ={-},
pages = {3-16},
year = {2017}
}
```

Same thing for the ris option. It works for different urls.

In [9]:
```
for doi in dois:
    w = Works(doi)
    print(w.ris())
```

```
TY  - JOUR
AU  - John R. Kitchin
PY  - 2015
TI  - Examples of Effective Data Sharing in Scientific Publishing
JO  - ACS Catalysis
VL  - 5
IS  - 6
SP  - 3894
EP  - 3899
DO  - https://doi.org/10.1021/acscatal.5b00538
ER  -
TY  - JOUR
AU  - Joshua Gopeesingh
AU  - M. Alexander Ardagh
AU  - Manish Shetty
AU  - Sean Burke
AU  - Paul J. Dauenhauer
AU  - Omar A. Abdelrahman
PY  - 2020
TI  - Resonance-Promoted Formic Acid Oxidation via Dynamic Electrocatalytic Modulation
```

```
JO  - ACS Catalysis
VL  - 10
IS  - 17
SP  - 9932
EP  - 9942
DO  - https://doi.org/10.1021/acscatal.0c02201
ER  -
TY  - JOUR
AU  - Yixin H. Ye
AU  - Ignacio E. Grossmann
AU  - José Carlos Pinto
PY  - 2017
TI  - Mixed-integer nonlinear programming models for optimal design of reliable chemical
plants
JO  - Computers & Chemical Engineering
VL  - 116
SP  - 3
EP  - 16
DO  - https://doi.org/10.1016/j.compchemeng.2017.08.013
ER  -
```

Show that the commandline utility works.

Run the command you created and show that it outputs either RIS or bibtex for a DOI.

The commandline command was created as an entry point in the `setup.py` file, and the `terminal_command` has the `@click` commands and definition of flags so one can run the cell below to print the reference in the terminal.

In [10]:
```
! ~/.local/bin/commandline --bibtex "https://doi.org/10.1021/acscatal.9b01606"
```

```
@article{M2019,
author = {M. Alexander Ardagh, Omar A. Abdelrahman and Paul J. Dauenhauer},
journal ={ACS Catalysis},
title = {Principles of Dynamic Heterogeneous Catalysis: Surface Resonance and Turnover F
requency Response},
volume = {9},
issue ={8},
pages = {6929-6937},
year = {2019}
}
```

In [11]:
```
! ~/.local/bin/commandline --ris "https://doi.org/10.1021/acscatal.9b01606"
```

```
TY  - JOUR
AU  - M. Alexander Ardagh
AU  - Omar A. Abdelrahman
AU  - Paul J. Dauenhauer
PY  - 2019
TI  - Principles of Dynamic Heterogeneous Catalysis: Surface Resonance and Turnover Freq
uency Response
JO  - ACS Catalysis
VL  - 9
IS  - 8
SP  - 6929
EP  - 6937
DO  - https://doi.org/10.1021/acscatal.9b01606
ER  -
```

In [12]:
```
# Run this cell to generate a pdf from this notebook
# Click the generated links to preview and download it.
# Report errors to Professor Kitchin
```

```
from s23 import pdf
%pdf
```

```
using webpdf
```

Open s23project/project_final.pdf

from webpdf

download s23project/project_final.pdf