Generated at Sat Apr 29 02:45:06 2023.

Source at https://jupyterhub-dev.cheme.cmu.edu/user/ziangliu@andrew.cmu.edu/lab/tree/s23-06682/assignments/project/project.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 = "Ziang Liu"
```

Project

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:

In [2]:
```
%%bash
# Clean up any existing files
rm -rf pkg
pip uninstall -y s23openalex
```
```
Found existing installation: s23openalex 0.0.1
Uninstalling s23openalex-0.0.1:
  Successfully uninstalled s23openalex-0.0.1
```

In [3]:
```
%%bash
mkdir -p pkg/s23openalex
cd pkg
git init
```
```
Initialized empty Git repository in /home/jupyter-ziangliu@andrew.cm-5e81f/s23-06682/ass
ignments/project/pkg/.git/
```

In [4]:
```
%%writefile pkg/setup.py
"""This file help to setup."""
```

```
from setuptools import setup

setup(
    name="s23openalex",
    version="0.0.1",
    description="bibtex and RIS",
    maintainer="Ziang Liu",
    maintainer_email="ziangliu@andrew.cmu.edu",
    license="MIT",
    packages=["s23openalex"],
    scripts=[],
    long_description="""get an RIS and a bibtex entry for a DOI""",
)
```

```
Writing pkg/setup.py
```

In [5]:

```
%%writefile pkg/s23openalex/works.py
"""This file could get an RIS and a bibtex entry for a DOI."""

import requests
import bibtexparser


class Works:
    """This class could get an RIS and a bibtex entry for a DOI."""

    def __init__(self, oaid):
        """Get an RIS and a bibtex entry for a DOI."""
        self.oaid = oaid
        self.req = requests.get(f"https://api.openalex.org/works/{oaid}")
        self.data = self.req.json()

    def get_bibtex(self):
        """Get a bibtex entry for a DOI."""
        h = "application/x-bibtex"
        res = requests.get(self.data["doi"], headers={"Accept": h})
        db = bibtexparser.loads(res.text)
        self.bibtex = db.entries[0]
        return self.bibtex

    def get_RIS(self):
        """Get an RIS for a DOI."""
        fields = []
        if self.data["type"] == "journal-article":
            fields += ["TY  - JOUR"]
        else:
            raise Exception("Unsupported type {self.data['type']}")

        for author in self.data["authorships"]:
            fields += [f'AU  - {author["author"]["display_name"]}']

        fields += [f'PY  - {self.data["publication_year"]}']
        fields += [f'TI  - {self.data["title"]}']
        fields += [f'JO  - {self.data["host_venue"]["display_name"]}']
        fields += [f'VL  - {self.data["biblio"]["volume"]}']

        if self.data["biblio"]["issue"]:
            fields += [f'IS  - {self.data["biblio"]["issue"]}']

        fields += [f'SP  - {self.data["biblio"]["first_page"]}']
        fields += [f'EP  - {self.data["biblio"]["last_page"]}']
        fields += [f'DO  - {self.data["doi"]}']
        fields += ["ER  -"]

        self.ris = fields
        return self.ris
```

```
Writing pkg/s23openalex/works.py
```

In [6]:
```
%%writefile pkg/s23openalex/__main__.py
"""This file use cmd line to get an RIS and a bibtex entry for a DOI."""

import argparse
from s23openalex import Works


def main():
    """Get an RIS and a bibtex entry for a DOI."""
    parser = argparse.ArgumentParser(description="Get RIS or bibtex entry.")
    parser.add_argument("doi", help="Input the DOI.")
    parser.add_argument("--RIS", help="Get the RIS.", action="store_true")
    parser.add_argument("--bib", help="Get the bibtex.", action="store_true")

    args = parser.parse_args()
    ww = Works(args.doi)
    if args.RIS:
        print(ww.get_RIS())
    elif args.bibtex:
        print(ww.get_bibtex())
    else:
        print("Please select an option (--RIS or --bibtex).")


if __name__ == "__main__":
    main()
```

```
Writing pkg/s23openalex/__main__.py
```

In [7]:
```
%%writefile pkg/s23openalex/__init__.py
"""This file start the pkg."""

from .works import Works
```

```
Writing pkg/s23openalex/__init__.py
```

In [8]:
```
%%writefile pkg/s23openalex/test_works.py
"""This file test the works."""

import pytest
from s23openalex import Works

bib = {
    "journal": "{ACS} Catalysis",
    "title": "Examples of Effective Data Sharing in Scientific Publishing",
    "author": "John R. Kitchin",
    "pages": "3894--3899",
    "number": "6",
    "volume": "5",
    "publisher": "American Chemical Society ({ACS})",
    "month": "may",
    "year": "2015",
    "url": "https://doi.org/10.1021%2Facscatal.5b00538",
    "doi": "10.1021/acscatal.5b00538",
    "ENTRYTYPE": "article",
    "ID": "Kitchin_2015",
}

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  -",
]


@pytest.fixture()
def setup_bib():
    """Get a bibtex entry for a DOI."""
    return bib


class TestBib:
    """Test a bibtex entry for a DOI."""

    def test_Bib(self, setup_bib):
        """Test a bibtex entry for a DOI."""
        w = Works("https://doi.org/10.1021/acscatal.5b00538")
        bib = w.get_bibtex()
        assert bib == setup_bib


@pytest.fixture()
def setup_RIS():
    """Get an RIS for a DOI."""
    return RIS


class TestRIS:
    """Test an RIS for a DOI."""

    def test_RIS(self, setup_RIS):
        """Test an RIS for a DOI."""
        w = Works("https://doi.org/10.1021/acscatal.5b00538")
        RIS = w.get_RIS()
        assert RIS == setup_RIS
```

```
Writing pkg/s23openalex/test_works.py
```

In [9]:
```
%%bash
cd pkg
ls .git/hooks
```

```
applypatch-msg.sample
commit-msg.sample
fsmonitor-watchman.sample
post-update.sample
pre-applypatch.sample
pre-commit.sample
pre-merge-commit.sample
prepare-commit-msg.sample
pre-push.sample
pre-rebase.sample
pre-receive.sample
update.sample
```

In [10]:
```
%%writefile pkg/.git/hooks/pre-commit
#!/bin/bash
echo "running precommit in `pwd`"
exit 0
```

```
Writing pkg/.git/hooks/pre-commit
```

```
In [11]: %%bash
         chmod +x pkg/.git/hooks/pre-commit
```

```
In [12]: %%bash
         cd pkg
         git add *.py
         git commit -m "adding pyfiles"
```

```
running precommit in /home/jupyter-ziangliu@andrew.cm-5e81f/s23-06682/assignments/projec
t/pkg
[master (root-commit) 09f8d08] adding pyfiles
 1 file changed, 15 insertions(+)
 create mode 100644 setup.py
```

```
In [13]: ! black pkg
```

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

```
In [14]: # ! black pkg && flake8 --exclude package/build pkg && pylint --ignore build pkg && pyte
         # ! black pkg && flake8 pkg && pylint pkg && pytest pkg
         ! black pkg && flake8 pkg && pylint pkg
```

```
All done! □ □ □
5 files left unchanged.
pkg/s23openalex/__init__.py:3:1: F401 '.works.Works' imported but unused
pkg/s23openalex/__main__.py:8:1: D202 No blank lines allowed after function docstring
pkg/s23openalex/__main__.py:8:1: D401 First line should be in imperative mood; try rephr
asing
pkg/s23openalex/__main__.py:13:80: E501 line too long (80 > 79 characters)
```

%%writefile pkg/s23openalex/**init**.py from .works import Works

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
```

```
In [15]: # This should install the s23openalex package
         !pip install ./pkg
```

```
Defaulting to user installation because normal site-packages is not writeable
Processing ./pkg
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: s23openalex
  Building wheel for s23openalex (setup.py) ... done
  Created wheel for s23openalex: filename=s23openalex-0.0.1-py3-none-any.whl size=3344 s
ha256=0d1bf339e781478ac263591ebec915d502966fa357ff062b799db0023de9de99
  Stored in directory: /tmp/pip-ephem-wheel-cache-td0cfegp/wheels/a0/63/fe/330c167faff38
0d6feafcb5aed7af2fb1e123aeb5a729d6373
Successfully built s23openalex
Installing collected packages: s23openalex
Successfully installed s23openalex-0.0.1
```

```
In [16]: ! tree pkg
```

```
pkg
├── build
│   ├── bdist.linux-x86_64
│   └── lib
│       └── s23openalex
│           ├── __init__.py
│               ├── __main__.py
│               ├── test_works.py
│               └── works.py
├── s23openalex
│   ├── __init__.py
│   ├── __main__.py
│   ├── test_works.py
│   └── works.py
├── s23openalex.egg-info
│   ├── dependency_links.txt
│   ├── PKG-INFO
│   ├── SOURCES.txt
│   └── top_level.txt
└── setup.py

6 directories, 13 files
```

Show evidence that your repo passes black and pylint

In [17]:
```
!pylint --version
```

```
pylint 2.14.5
astroid 2.11.7
Python 3.9.7 | packaged by conda-forge | (default, Sep 29 2021, 19:20:46)
[GCC 9.4.0]
```

In [18]:
```
# %%bash
# black pkg
# pylint pkg

! black pkg && flake8 pkg && pylint pkg
```

```
All done! 
5 files left unchanged.
pkg/s23openalex/__init__.py:3:1: F401 '.works.Works' imported but unused
pkg/s23openalex/__main__.py:8:1: D202 No blank lines allowed after function docstring
pkg/s23openalex/__main__.py:8:1: D401 First line should be in imperative mood; try rephr
asing
pkg/s23openalex/__main__.py:13:80: E501 line too long (80 > 79 characters)
pkg/build/lib/s23openalex/__init__.py:3:1: F401 '.works.Works' imported but unused
pkg/build/lib/s23openalex/__main__.py:8:1: D202 No blank lines allowed after function do
cstring
pkg/build/lib/s23openalex/__main__.py:8:1: D401 First line should be in imperative mood;
 try rephrasing
pkg/build/lib/s23openalex/__main__.py:13:80: E501 line too long (80 > 79 characters)
```

In [ ]:

Tests

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

In [19]:
```
! pytest pkg
```

```
============================= test session starts ==============================
platform linux -- Python 3.9.7, pytest-7.2.2, pluggy-1.0.0
rootdir: /home/jupyter-ziangliu@andrew.cm-5e81f/s23-06682/assignments/project/pkg
plugins: typeguard-2.13.3, anyio-3.6.1
```

```
collected 2 items

pkg/s23openalex/test_works.py ..                                         [100%]

============================== 2 passed in 0.51s ===============================
```

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.

In [20]:
```python
import s23openalex
from s23openalex import Works
w1 = Works('https://doi.org/10.1088/2058-7058/20/11/28')
w1.get_bibtex()
```

Out[20]:
```
{'journal': 'Physics World',
 'title': 'Access all theses',
 'author': 'Michael Banks',
 'pages': '18--19',
 'number': '11',
 'volume': '20',
 'publisher': '{IOP} Publishing',
 'month': 'nov',
 'year': '2007',
 'url': 'https://doi.org/10.1088%2F2058-7058%2F20%2F11%2F28',
 'doi': '10.1088/2058-7058/20/11/28',
 'ENTRYTYPE': 'article',
 'ID': 'Banks_2007'}
```

In [21]:
```python
w1.get_RIS()
```

Out[21]:
```
['TY  - JOUR',
 'AU  - Michael Banks',
 'PY  - 2007',
 'TI  - Access all theses',
 'JO  - Physics World',
 'VL  - None',
 'SP  - None',
 'EP  - None',
 'DO  - https://doi.org/10.1088/2058-7058/20/11/28',
 'ER  -']
```

In [22]:
```python
w2 = Works('https://doi.org/10.12688/mniopenres.12772.1')
w2.get_bibtex()
```

Out[22]:
```
{'journal': '{MNI} Open Research',
 'title': 'From data sharing to data publishing',
 'author': 'Jean-Baptiste Poline',
 'pages': '1',
 'volume': '2',
 'publisher': 'F1000 Research Ltd',
 'month': 'jan',
 'year': '2018',
 'url': 'https://doi.org/10.12688%2Fmniopenres.12772.1',
 'doi': '10.12688/mniopenres.12772.1',
 'ENTRYTYPE': 'article',
 'ID': 'Poline_2018'}
```

In [23]:
```python
w2.get_RIS()
```

```
['TY  - JOUR',
Out[23]:  'AU  - Jean-Baptiste Poline',
          'PY  - 2018',
          'TI  - From data sharing to data publishing',
          'JO  - MNI open research',
          'VL  - 2',
          'SP  - 1',
          'EP  - 1',
          'DO  - https://doi.org/10.12688/mniopenres.12772.1',
          'ER  -']
```

Show that the commandline utility works.

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

```
In [24]: ! python -m s23openalex --RIS 'https://doi.org/10.1088/2058-7058/20/11/28'
```

```
['TY  - JOUR', 'AU  - Michael Banks', 'PY  - 2007', 'TI  - Access all theses', 'JO  - Ph
ysics World', 'VL  - None', 'SP  - None', 'EP  - None', 'DO  - https://doi.org/10.1088/2
058-7058/20/11/28', 'ER  -']
```

```
In [25]: ! python -m s23openalex --bibtex 'https://doi.org/10.1088/2058-7058/20/11/28'
```

```
{'journal': 'Physics World', 'title': 'Access all theses', 'author': 'Michael Banks', 'p
ages': '18--19', 'number': '11', 'volume': '20', 'publisher': '{IOP} Publishing', 'mont
h': 'nov', 'year': '2007', 'url': 'https://doi.org/10.1088%2F2058-7058%2F20%2F11%2F28',
'doi': '10.1088/2058-7058/20/11/28', 'ENTRYTYPE': 'article', 'ID': 'Banks_2007'}
```

```
In [26]: # 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 project.pdf

from webpdf

download project.pdf