Exceptions, packaging, Lisence

ÉCOLE **NORMALE**
SUPÉRIEURE
DE **LYON**

Ali Farnudi, Fall 2021

Exception Handeling in Python

Errors

- Bugs: Syntax and logical errors
- Runtime errors:
  - Cannot locate a file
  - Network is down
  - A token is expired
  - etc

Little bit tricky with Python since we are interpreting!

Hands on - Examples 0-3

Try

```
try:
    Some operations
except ExceptionI:
    If there is ExceptionI, then execute this block.
except ExceptionII:
    If there is ExceptionII, then execute this block.
else:
    If there is no exception then execute this block.
finally:
    This would always be executed.
```

Hands on - raise -exercises >= 4

What if a file is left open?

- What if you don’t close a file:
  - Your program -> python garbage collectors hands -> auto closed in theory (may not be closed)
  - Too many things open == slow down
  - Most file changes happen after file is closed
  - There is a limit on how many files you can have open
  - Some OS (like Windows) treat open files as locked

```
f = open('file', 'r')
read_data = f.read()
f.close()
```

```
with open('file', 'r') as f:
    read_data = f.read()
```

Software Packaging

Npm’s “Left-pad problem”

11 lines of code
downloaded 2.5 million times the moth before

Why PyPI?

- Copy & paste is **not** the way to share your code
- Developers can put tests and help improve the code
- Hopefully get developers depend on it

Publish code
make python better

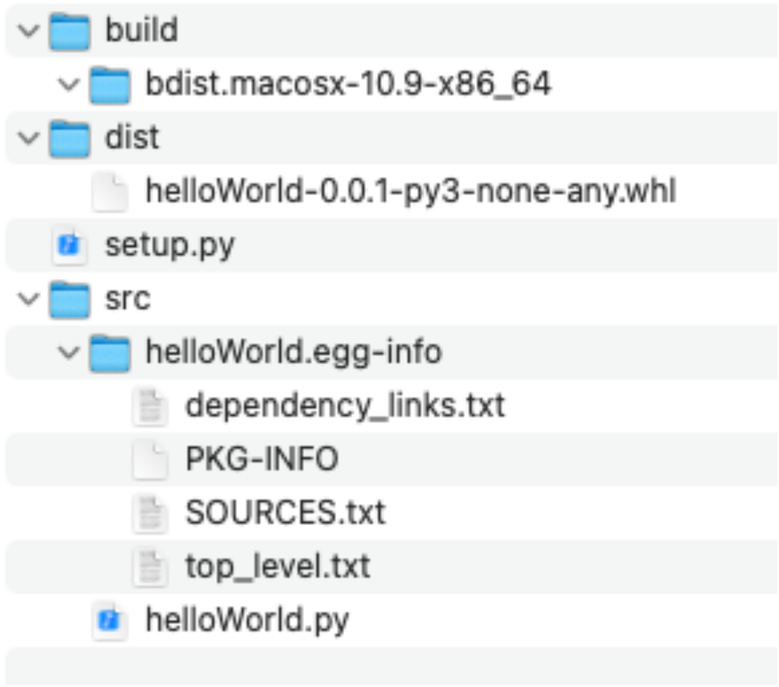
- General code that others can use
- Extract and make independent
- Put it in a src directory
- Make a setup.py file

```
def say_hello(name=None):
    if name is None:
        return "Hello, world!"
    else:
        return f"Hello, {name}!"
```

```python
from setuptools import setup

setup(
    name="helloWorld", #what you write after $ pip install
    version="0.0.1", # 0.0.x generaly means it is nstable
    description="Say hello!", # usually one liner
    py_modulies=["helloWorld"], # list of actual python modules (code)
    package_dir={"": "src"},
)
```

- $ python setup.py bdist_wheel
- Git ignore src/helloWorld.egg-info



Test package

Run every time setup.py changes

- $ pip install -e .
- -e : Link (not copy) to the library
- . : find package here

```
>>> from helloWorld import say_hello
>>> say_hello()
'Hello, world!'
>>> say_hello('Ali')
'Hello, Ali!'
>>>
```

Before uploading?

gitignore & classifiers

- gitignore.io
- License

- Add classifiers
  https://pypi.org/classifiers/

```
    classifiers=[
        "Programming Language :: Python :: 3",
        "Programming Language :: Python :: 3.6",
        "Programming Language :: Python :: 3.7",
        "License :: OSI Approved :: GNU General Public License v3 (GPLv3)",
        "Operating System :: OS Independent",
    ],
)
```

Before uploading?

Documentation

```
from setuptools import setup

with open("README.rst", "r") as fh:
    long_description = fh.read()

setup(
    name="helloWorld", #what you write after $ pip install
    version="0.0.1", # 0.0.x generaly means it is nstable
    description="Say hello!", # usually one liner

    long_description=long_description,
    long_description_content_type="text/x-rst",
```

README.rst or
README.md

Before uploading?

Dependencies

- Make sure restrictions are as relaxed as they can be

- $ pip install -e .

```
from setuptools import setup

with open("README.rst", "r") as fh:
    long_description = fh.read()

setup(
    name="helloWorld", #what you write after $ pip install
    version="0.0.1", # 0.0.x generaly means it is nstable
    description="Say hello!", # usually one liner
    long_description=long_description,
    long_description_content_type="text/x-rst",
    py_modulies=["helloWorld"], # list of actual python modu
    package_dir={"": "src"},
    classifiers=[
        "Programming Language :: Python :: 3",
        "Programming Language :: Python :: 3.6",
        "Programming Language :: Python :: 3.7",
        "License :: OSI Approved :: GNU General Public Licen
        "Operating System :: OS Independent",
    ],
    install_requires = [
        "package ~=1.7",
    ],
)
```

```
    py_modulies=["helloWorld"],
    package_dir={"": "src"},
    classifiers=[
        "Programming Language :: Python ...
        "Programming Language :: Python ::...
        "Programming Language :: Python :: ...
        "License :: OSI Approved :: GNU General...
        "Operating System :: OS Independent",
    ],
    install_requires = [
        "numpy ~=1.7",
    ],

    extras_require = {
        "dev":[
            "pytest>=3.7",
        ],
    },
)
```

Tests

pyTest

- **Development** dependencies
- Update README

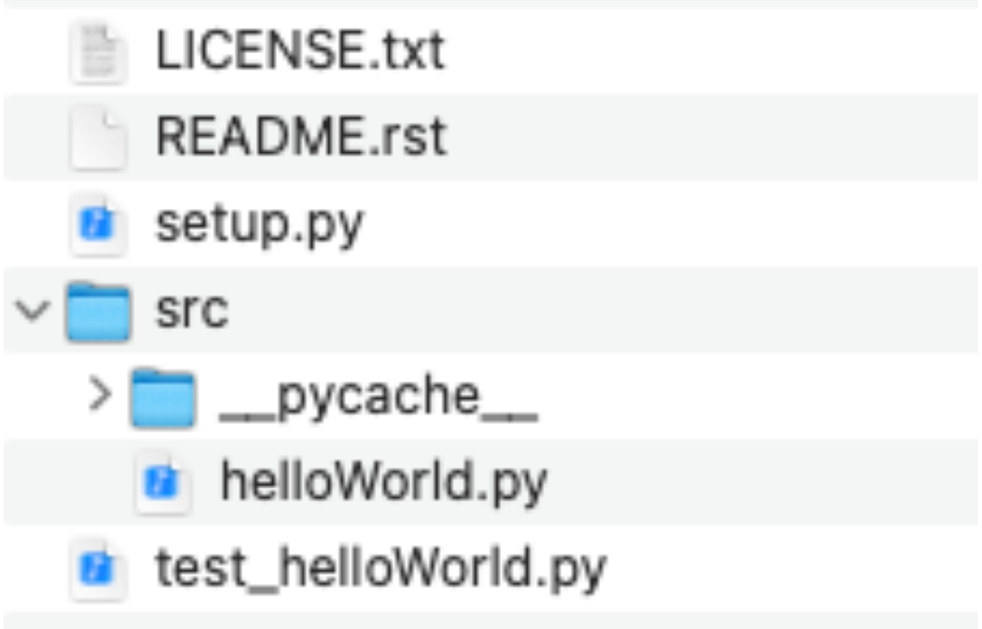
Almost done!

```
version="0.0.1", # 0.0.x generaly means it i
description="Say hello!", # usually one line
long_description=long_description,
long_description_content_type="text/x-rst",
py_modulies=["helloWorld"], # list of actual
package_dir={"": "src"},

url="https://github.com/...",
auther="name surname",
auther_email="user@example.com",

classifiers=[
    "Programming Language :: Python :: 3",
    "Programming Language :: Python :: 3.6",
    "Programming Language :: Python :: 3.7",
    "License :: OSI Approved :: GNU General
    "Operating System :: OS Independent",
],
install_requires = [
    "numpy ~=1.7",
],
```

Check distribution



- $ python setup.py sdist

- $ tar tzf dist/helloWorld-0.0.1.tar.gz
  - Check everything
  - Use “check-manifest” If you want to include other stuff in package:

Publish!

As soon as you have covered the basics

- $ python setup.py bdist_wheel sdist
- $ ls dist
  - helloWorld-0.0.1-py3-none-any.whl
  - helloWorld-0.0.1.tar.gz
- Push to PyPI with **twine**:
  - $ pip install twine
  - twine upload dist/*
    - Asks

Go to PyPI website to check your package

Is there a shortcut?

- $ pip install cookicutter
- $ cookicutter gh:ionelmc:cookicutter-pylibrary

Licensing

Who has used Open Source software?

- Softwares usually come with a license
- It will determine what a user can and cannot do with the software
- Some licenses are free, some you have to purchase, some might have different requirements.
- Licenses are written by lawyers and not by developers.
- But developers must know the basics.
- Facebook’s ReactNative vs Apache

General License Types

- Permissive
  - Users have more freedom
- Copyleft
  - Makes sure the software stays on the same license

Permissive

- Fewer restrictions on what users can do with the software
- Allows companies to use the software without contributing back
- Apache, MIT, VSD, creative commons attribution
  - Sharing material within specific parameters.
    - Crediting authors for their work
  - Allows free legal copying/distribution/sharing of content
  - Like what Samsung does with Android.
    - Most of Android in on Apache license but Samsung freely modifies it and does not release the changes. And sell it

Copyleft

Want to keep it on the license

- If you use it you have to give back to the community
- Prevents companies from using the software without contributing back.
- Can have potential licensing conflicts.
- How linux Kernel is setup underneath Android
- GPL, CC attribute share.., IBM, Mozilla

Which license should I choose?

- Depends on your project.
  - If you want to write a library so that anyone can take it and put it in their software and use it how they want, better off with permissive (Apache)

Dual licensing

- Usually it means the user can pick one license
  - Your code must be compatible with both licenses
- Occasionally projects require users to abide by both licenses.

Contributor License Agreement (CLA)

- Gives project leader the ability to relicense the codebase.
- Good: Allows projects to adapt as the licensing situation changes.
- Bad: Allows projects to be dramatically shifted away from the contributors wishes.
- Bad/Good: Companies can take the code and relicense and sell it to other companies without contributing (Canonical, Eclipse, Cyanogen, etc)

What if you don’t choose a license?

- exclusive copyright by default
  - No one can copy, distribute, or modify
  - Risk of take-downs, shake-downs, or litigation
- If you publish your source code on GitHub
  - Github’s Terms of Service
  - Allow others to view and fork your repository

What if you find a code without a license

- Ask the maintainers nicely to add a license.
  - Don’t use the software. Find alternatives
  - If on GitHub, open an issue requesting a license
  - Negotiate a private license (Bring your lawyer).

https://choosealicense.com