

Improving Your Python Programming Experience

**Tools and Tricks to
Make Your Life Easier**

**Jacob Adams, UGRC
UGIC 2022**



2019 SHOP TOUR



Rockville

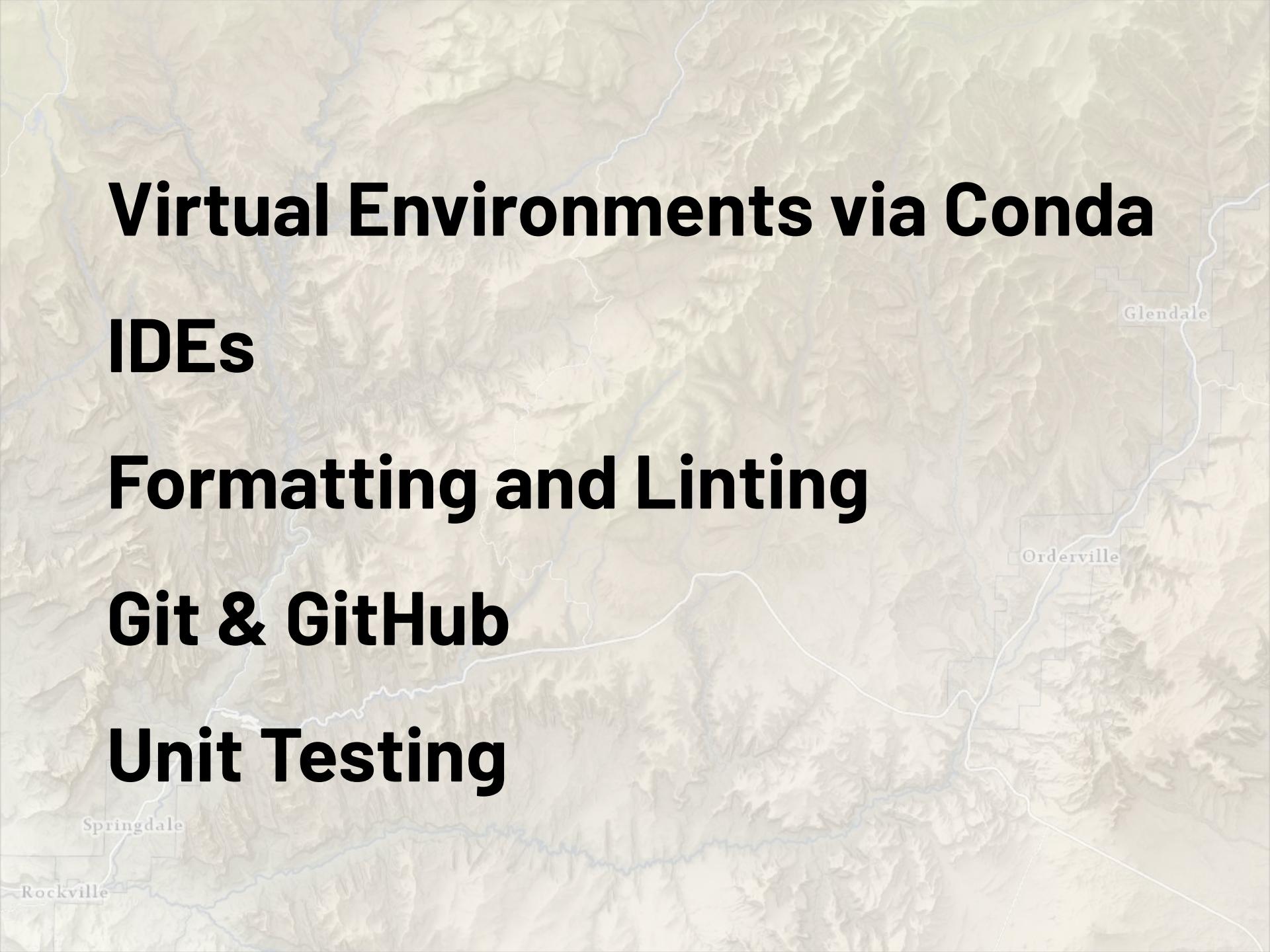
Virtual Environments via Conda

IDEs

Formatting and Linting

Git & GitHub

Unit Testing





**DON'T
PANIC**

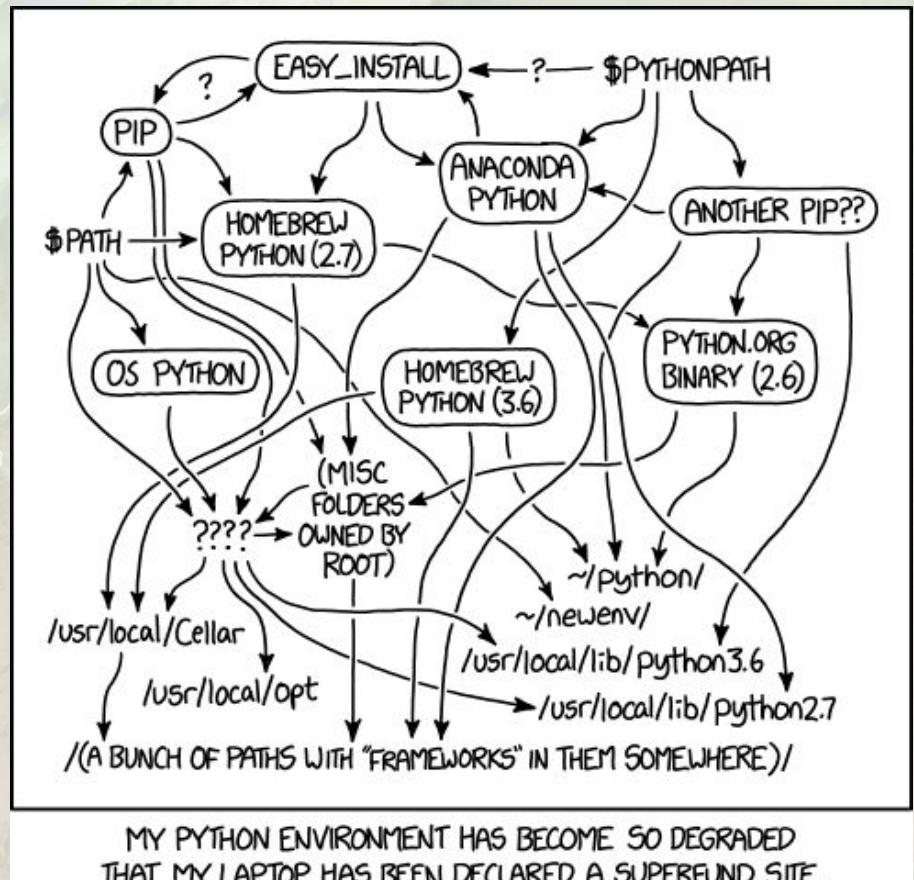
Springdale

Rockville

Glendale

Orderville

Managing Your Environments With CONDA



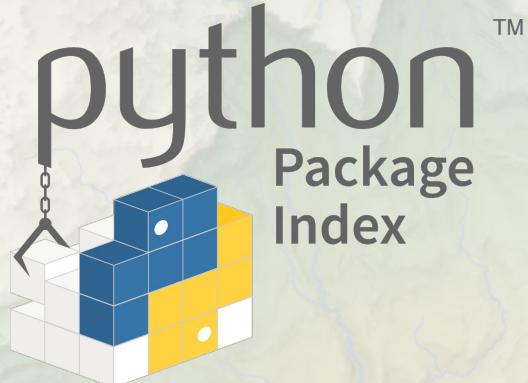
```
(arcgispro-py3) c:\gis>conda env list
# conda environments:
#
base          C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python
arcgispro-py3 * C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\arcgispro-py3
auditor        C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\auditor
erapskid       C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\erapskid
gsheets        C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\gsheets
housing        C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\housing
palletjack     C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\palletjack
rural          C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\rural
supervisor     C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\supervisor
test           C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\test
ugic           C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\ugic
upython        C:\Users\jdadams\AppData\Local\Programs\ArcGIS\Pro\bin\Python\envs\upython
```

Conda virtual environments: separate sandboxes for your projects



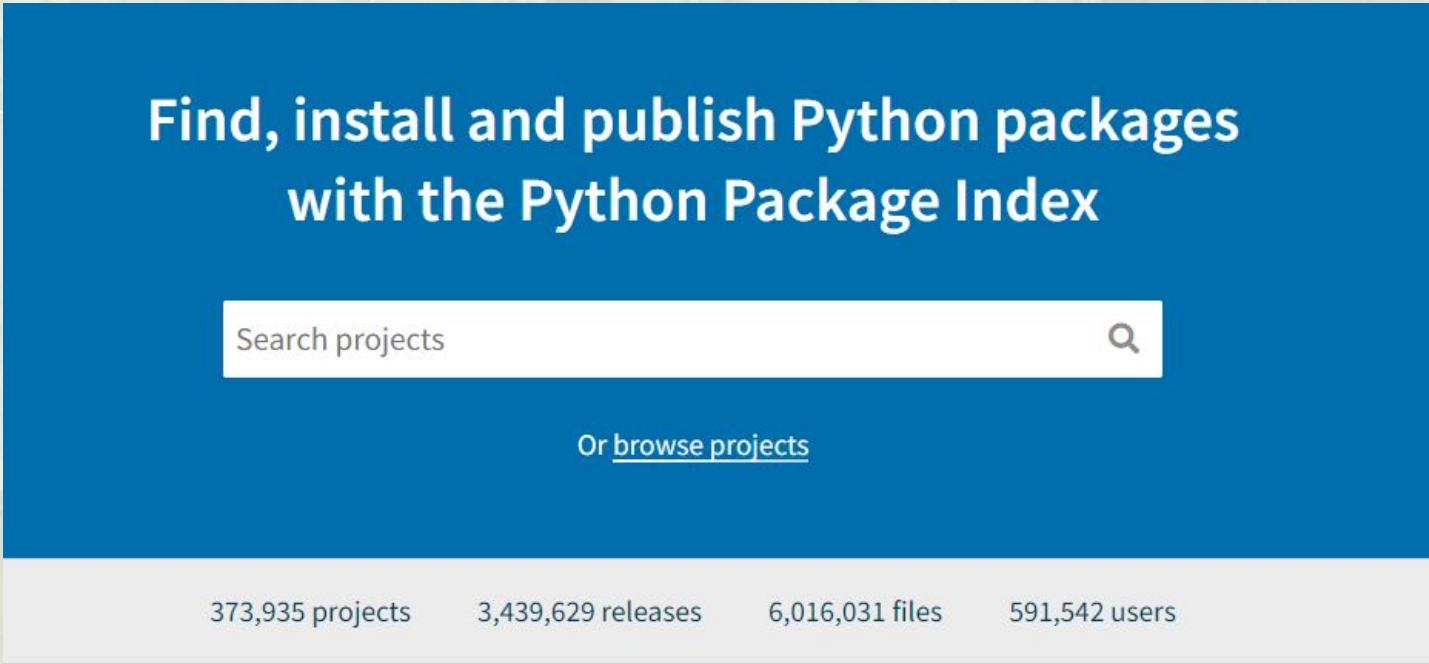
Lowes.com

CONDA vs



- Installs Python
- Many channels
- Manages environments
- Installed with ArcGIS Pro
- Requires Python
- More packages
- Easy to automate dependencies
- Easily install local packages

Packaging Your Code

A screenshot of the Python Package Index (PyPi) homepage. The background is a light blue color with a faint map of Colorado and the city of Boulder. The main title "Find, install and publish Python packages with the Python Package Index" is centered in white text. Below it is a search bar with the placeholder "Search projects" and a magnifying glass icon. Underneath the search bar is a link "Or [browse projects](#)". At the bottom of the page, there is a white footer bar with four pieces of information: "373,935 projects", "3,439,629 releases", "6,016,031 files", and "591,542 users".

Find, install and publish Python packages
with the Python Package Index

Search projects

Or [browse projects](#)

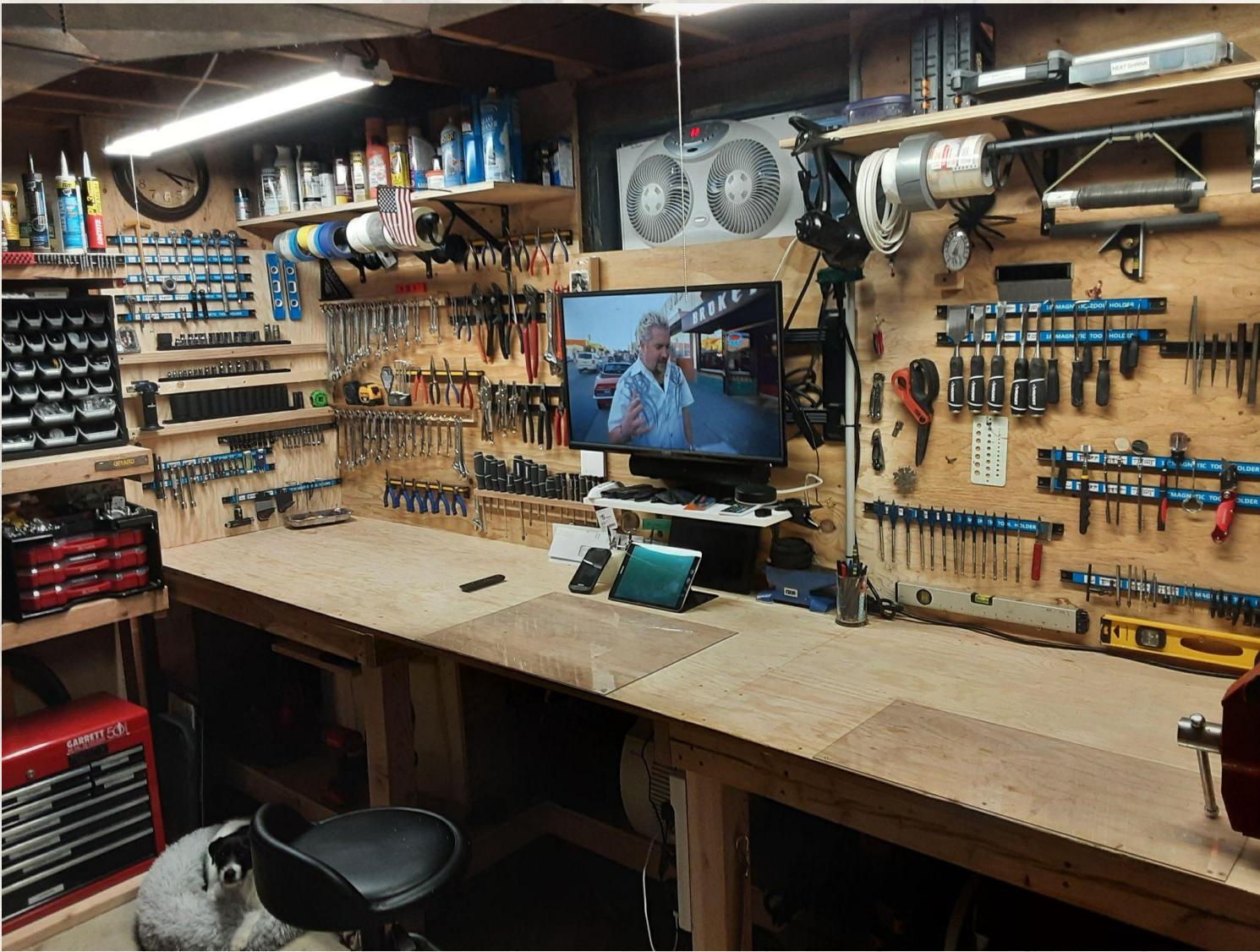
373,935 projects 3,439,629 releases 6,016,031 files 591,542 users

Configuration: `setup.py`

Local Installation: `pip install -e . [tests]`

Distribution: Upload to PyPi

Writing with an IDE



https://www.reddit.com/r/Workbenches/comments/lsl4v0/small_but_gets_it_done/

Visual Studio Code:

ONE IDE



TO RULE THEM ALL

memegenerator.net

- **Code hinting**
 - **Highlighting**
 - **Multi-select**
 - **Go to definition**
 - **Extensions**
 - **Renaming**
 - **Debugging**

```
File Edit Selection View Go Run Terminal Help

EXPLORER ... + setup.py + updaters.py 8 + test_updaters.py

PALLETJACK ...
    > __pycache__
    > .eggs
    > .github
    > .pytest_cache
    > .vscode
    > build
    > dist
    > docs
        + src
            + palletjack
                > __pycache__
                + __init__.py
                + loaders.py
                + updaters.py 8
                > ugrc_palletjack.egg-info
            + tests
                > __pycache__
                + mock arcpy.py
                + test_loaders.py
                + test_updaters.py
        .coverage
        .coveragerc
        .editorconfig
        .gitignore
        cov.xml
        LICENSE
        pylintrc
        README.md
        setup.cfg
        setup.py

src > palletjack > updaters.py > FeatureServiceAttachments
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336

def _overwrite_attachments_by_oid(overwrites_dict):
    overwrites_count = 0
    for row in overwrites_dict.values():
        target_oid = row['OBJECTID']
        filepath = row[attachment_id]
        attachment_id = row['ID']
        old_name = row['NAME']

        self._class_logger.debug('Overwriting %s (%s)' % (filepath, target_oid))
        result = self.feature_layer.updateAttachment(filepath, target_oid)
        self._class_logger.debug(result)
        if not result['updateAttachment']:
            warnings.warn('Failed to update attachment %s' % target_oid)
            continue
        overwrites_count += 1
    return overwrites_count

def update_attachments(self, feature_layer_itemid, live_features_as_df):
    self._class_logger.info('Updating attachments')
    self._class_logger.debug('Using itemid %s' % feature_layer_itemid)
    self.feature_layer = self.gis
    live_data_subset_df = pd.DataFrame(live_features_as_df)
```

Formatting & Linting



Why Code Style Matters



**THIS IS NOT THE
CODE STYLE YOU ARE LOOKING
FOR**

memegenerator.net

Green River

Formatting vs Linting

- [PEP 8](#)
- Consistent style
- Doc strings and comments
- Performed by yapf
- Controlled by config file
`(setup.cfg, pyproject.toml, ...)`

- Detects errors and "code smell"
- Enforces good programming practices
- Performed by pylint
- Controlled by `.pylintrc`

Getting a Handle on File Names: Version Control with Git & GitHub

JORGE CHAM © 2012

"FINAL".doc



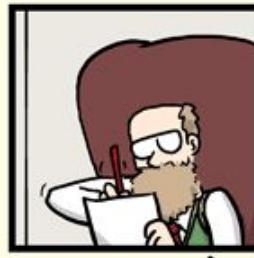
FINAL.doc!



FINAL_rev.2.doc



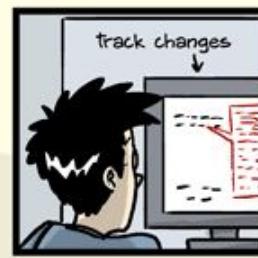
FINAL_rev.2.doc



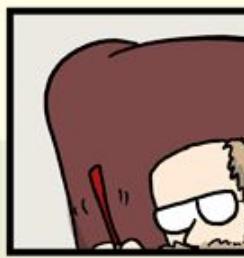
FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



FINAL_rev.18.comments7.
corrections9.MORE.30.doc



FINAL_rev.22.comments49.
corrections.10.#@\$%WHYDID
ICOMETOGRAD SCHOOL?????.doc



THE MOMENT YOU REALIZE



THE TRUE MEANING OF MEASURE
TWICE, CUT ONCE.

generator.net

Kingston



- Solves the `_v3_try6.py` problem
- Version control
- Branching
- CLI and desktop clients
- Distributed repositories



- Online home for repositories
- Pull Requests
- Issue tracking
- Readmes and documentation
- Continuous Integration (CI)
- A whole lot more

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.

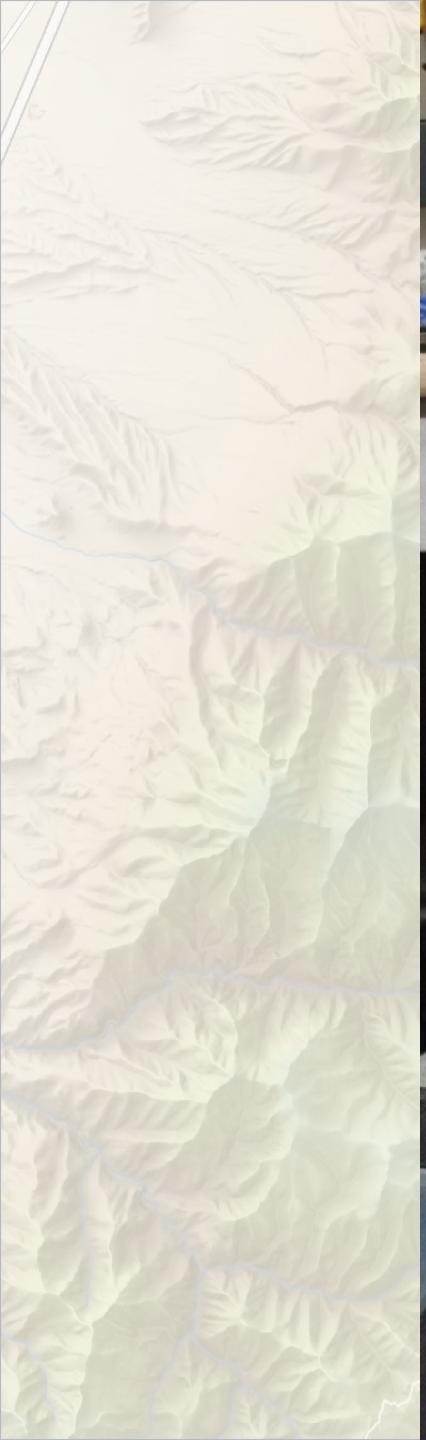
COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.



Unit Testing: Knowing When You Break Things





<https://reddit.com/u/Patchen35>

Benefits of Unit Tests

- Trusting your code
- Understanding your code
- Catching errors faster
 - Especially when updating
- Catching more edge cases
- Writing better code

Good Program Design

The screenshot shows a code editor's outline panel with a dark theme. It lists several functions under the 'OUTLINE' section:

- > `get_proper_built_yr_value_series`
- > `change_geometry`
- > `add_extra_info_from_csv` 1
- > `get_centroids_copy_of_polygon_df`
- > `load_and_clean_parcels`
- > `clean_dissolve_field_names`
- > `get_non_base_addr_points` ●
- > `set_common_area_types`
- > `subset_owned_unit_groupings_from_comm...`
- > `set_multi_family_single_parcel_subtypes` 2
- > `get_address_point_count_series`
- > `standardize_fields`
- > `concat_evaluated_dataframes` ●
- > `classify_from_area` 3
- > `get_common_areas_intersecting_parcel...` 2
- > `concat_cities_metro_townships`

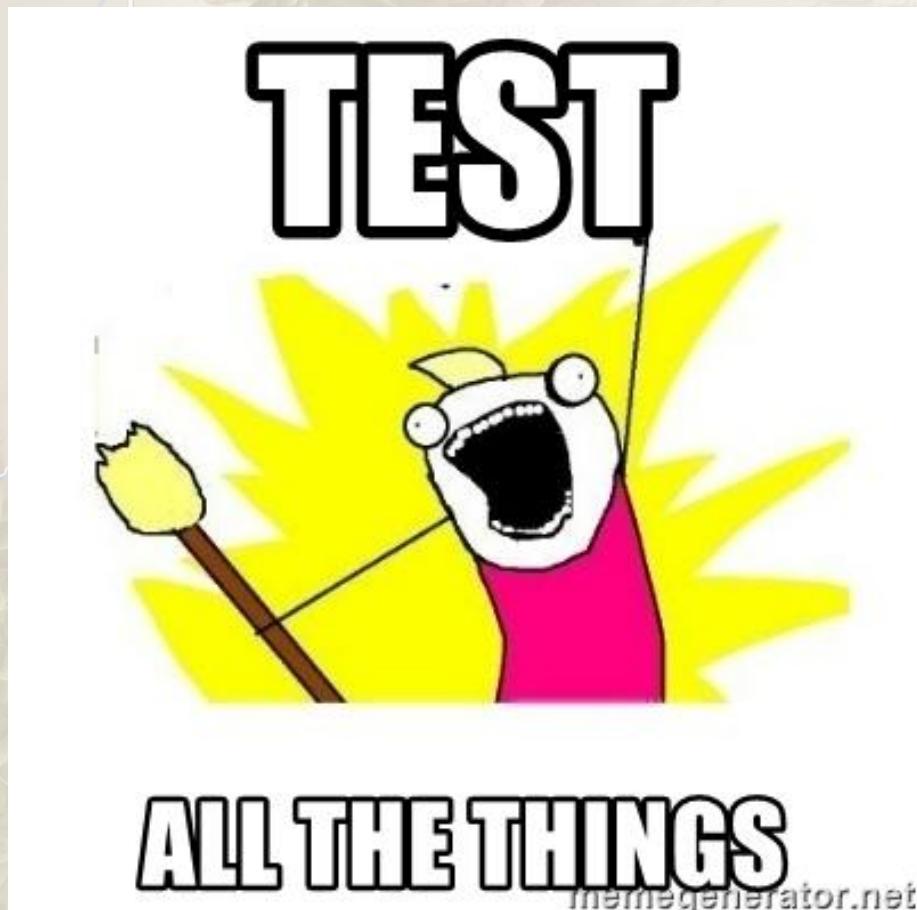
Bad Program Design

The screenshot shows a code editor's outline panel with a dark theme. It lists several functions and variables under the 'OUTLINE' section, including a 'Chunk' node:

- > `Chunk`
- > `sizeof_fmt` 2
- > `stretch_scale` ●
- > `ProcessSuperArray` 8
- > `lock_init` 2
- [●] `lock`
- > `ParallelRCP` +9
- [●] `args`
- [●] `all` 1
- [●] `kernel_args`
- [●] `blur_gauss_args`
- [●] `mdenoise_args`
- [●] `clahe_args`
- [●] `hs_args`
- [●] `sky_args`
- [●] `out_args`
- [●] `arguments`
- [●] `arg_dict`
- [●] `input DEM`

Basic Steps of a Unit Test

- Arrange
- Act
- Assert



memegenerator.net

Unit Test Tools

`unittest`



`pytest-mock`

`pytest-cov`



UGRC

Utah Geospatial Resource Center

jadams@utah.gov

gis.utah.gov/presentations

Hideout

Oakley

Kamas

Resources

- **UGRC Python template GitHub repo**
 - <https://github.com/agrc/python>
- **Conda docs**
 - <https://docs.conda.io/en/latest/>
- **pip docs**
 - <https://pip.pypa.io/en/stable/>
- **Virtual Studio Code**
 - <https://code.visualstudio.com/docs>
- **pylint docs**
 - <https://pylint.pycqa.org/en/latest/>
- **yapf docs**
 - <https://github.com/google/yapf>

Resources

- **Git docs**
 - <https://git-scm.com/doc>
- **Some decent git tutorials**
 - <https://www.atlassian.com/git/tutorials>
- **GitHub docs**
 - <https://docs.github.com/en>
- **unittest.mock reference**
 - <https://docs.python.org/3/library/unittest.mock.html>
- **pytest docs**
 - <https://docs.pytest.org/en/6.2.x/contents.html>
- **pytest-mock docs**
 - <https://pypi.org/project/pytest-mock/>
- **Some patchy notes on testing**
 - <https://github.com/jacobdadam.../blob/master/pytest.md>