

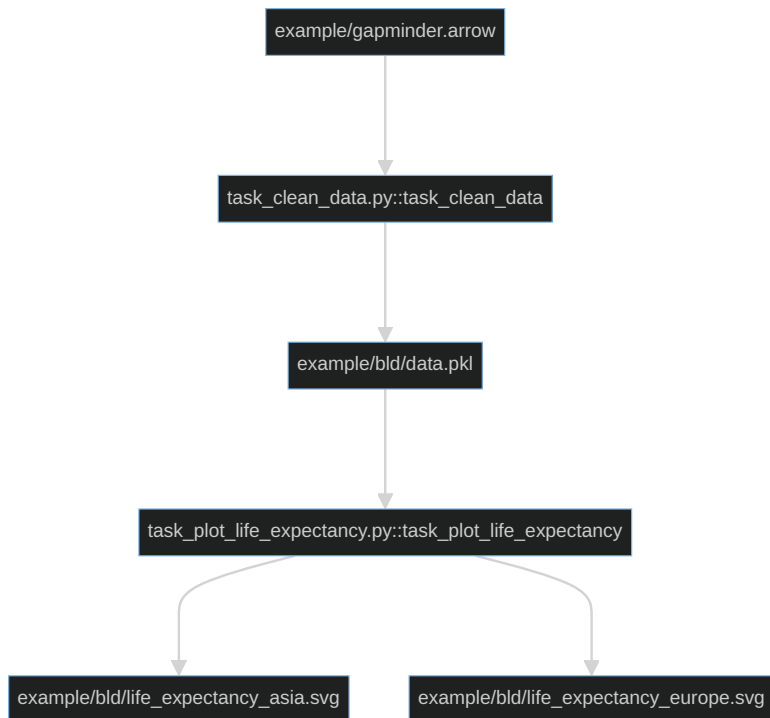
Effective Programming Practices for Economists

Reproducible Research

Writing (py)tasks with multiple outputs

Janoś Gabler and Hans-Martin von Gaudecker

Tiny example, extended



- How do we tell pytask that we have two products for `task_plot_life_expectancy`?
- How would we tell pytask that we had more than one dependency?

Contents of task_clean_data.py

```
from pathlib import Path
```

```
import pandas as pd
```

```
BLD = Path(__file__).parent / "bld"
```

```
def task_clean_data(raw_file=Path("gapminder.arrow"), produces=BLD / "data.pkl"):  
    raw = pd.read_feather(raw_file)  
    clean = _clean_data(raw)  
    clean.to_pickle(produces)
```

```
def _clean_data(raw):  
    df = raw.rename(  
        columns={  
            "lifeExp": "life_exp",  
            "gdpPercap": "gdp_per_cap",  
        },
```

Contents of task_plot_life_expectancy.py

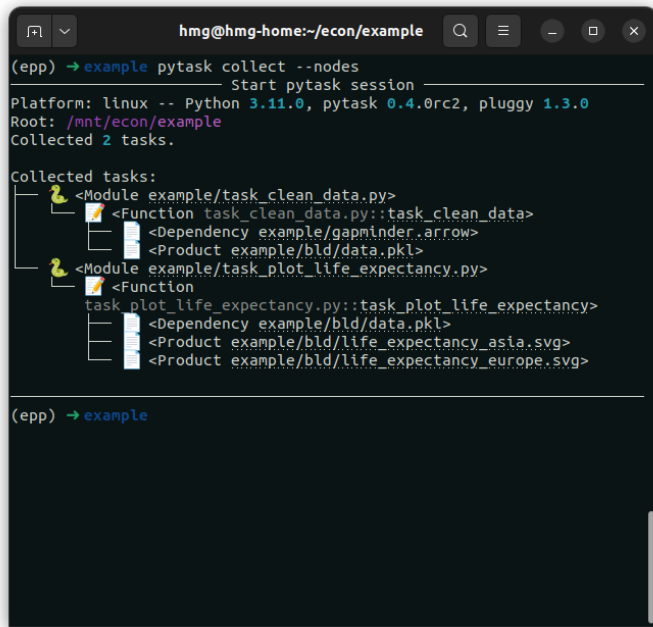
```
BLD = Path(__file__).parent / "bld"

products = {
    "Asia": BLD / "life_expectancy_asia.svg",
    "Europe": BLD / "life_expectancy_europe.svg"
}

def task_plot_life_expectancy(
    data_file=BLD / "data.pkl",
    produces=products,
):
    df = pd.read_pickle(data_file)
    for region, fig_file in produces.items():
        fig = _plot_life_expectancy(df[df["continent"] == region])
        fig.write_image(fig_file)
```

Verify Dependency graph (DAG, tree)

- Inspect function signatures to build a dependency graph
- Both values of `products` dict passed to `produces` argument have become nodes!

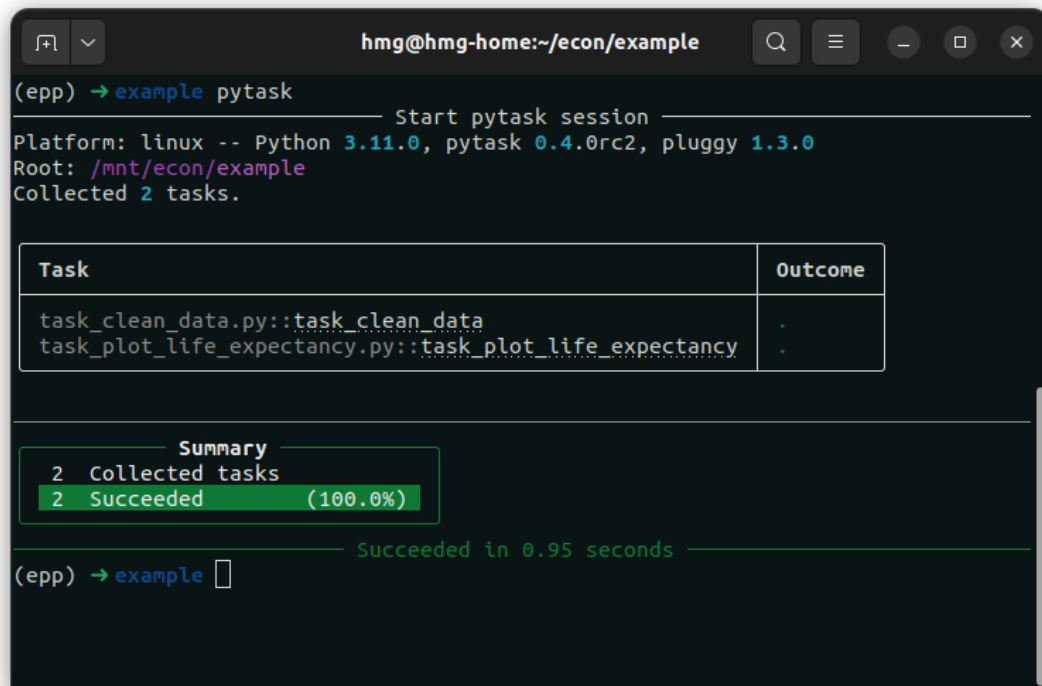
A terminal window with a dark background and light text. The title bar shows 'hmg@hmg-home:~/econ/example'. The prompt is '(epp) → example'. The command 'pytask collect --nodes' has been executed. The output shows the start of a pytask session on a Linux platform with Python 3.11.0, pytask 0.4.0rc2, and pluggy 1.3.0. The root directory is '/mnt/econ/example' and 2 tasks were collected. The collected tasks are listed in a tree structure. The first task is from 'example/task_clean_data.py' and contains a function 'task_clean_data' with a dependency on 'example/gapminder.arrow' and a product 'example/bld/data.pkl'. The second task is from 'example/task_plot_life_expectancy.py' and contains a function 'task_plot_life_expectancy' with a dependency on 'example/bld/data.pkl' and two products: 'example/bld/life_expectancy_asia.svg' and 'example/bld/life_expectancy_europe.svg'. The prompt at the bottom is '(epp) → example'.

```
hmg@hmg-home:~/econ/example
(epp) → example pytask collect --nodes
Start pytask session
Platform: linux -- Python 3.11.0, pytask 0.4.0rc2, pluggy 1.3.0
Root: /mnt/econ/example
Collected 2 tasks.

Collected tasks:
└─ <Module example/task_clean_data.py>
   └─ <Function task_clean_data.py::task_clean_data>
      └─ <Dependency example/gapminder.arrow>
         └─ <Product example/bld/data.pkl>
└─ <Module example/task_plot_life_expectancy.py>
   └─ <Function task_plot_life_expectancy.py::task_plot_life_expectancy>
      └─ <Dependency example/bld/data.pkl>
         └─ <Product example/bld/life_expectancy_asia.svg>
            └─ <Product example/bld/life_expectancy_europe.svg>

(epp) → example
```

Run pytask

A terminal window with a dark background. The title bar shows the user 'hmg' at 'hmg-home' in the directory '~/econ/example'. The prompt is '(epp) → example'. The command 'pytask' has been executed. The output shows the start of a pytask session, platform and version information, and a list of collected tasks. A table displays the execution of two tasks, both of which succeeded. A summary box highlights that 2 tasks were collected and 2 succeeded (100.0%). The total execution time is 0.95 seconds. The prompt is now '(epp) → example' followed by a cursor.

```
hmg@hmg-home:~/econ/example
(epp) → example pytask

Start pytask session

Platform: linux -- Python 3.11.0, pytask 0.4.0rc2, pluggy 1.3.0
Root: /mnt/econ/example
Collected 2 tasks.



| Task                                                    | Outcome |
|---------------------------------------------------------|---------|
| task_clean_data.py::task_clean_data                     | ✓       |
| task_plot_life_expectancy.py::task_plot_life_expectancy | ✓       |



Summary
2 Collected tasks
2 Succeeded (100.0%)

Succeeded in 0.95 seconds

(epp) → example
```

Multiple dependencies and products

- Defaults to keyword arguments may hold
 - a single `pathlib.Path`
 - a container of `pathlib.Path` objects
 - Container may be nested, so long as the atomic elements are `pathlib.Path` objects
- For dependencies, can pass as many different arguments with defaults as you like