

Software Testing Final Project

TDD compose-viz

Team 14

吳星翰 / 林奕宏 / 歐陽詮 / 王均琦 / 施泓丞

TABLE OF CONTENTS

1. About compose-viz

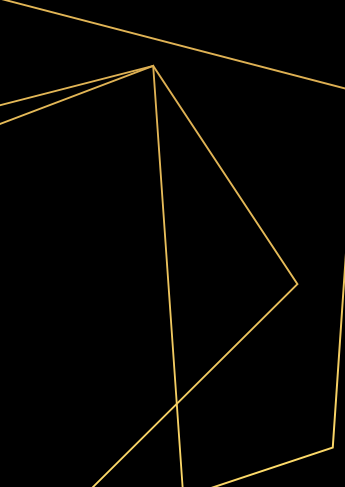
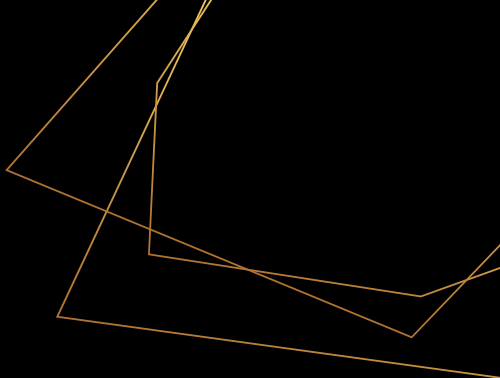
2. How it works

3. Testing

4. Pre-commit

5. Experience Sharing

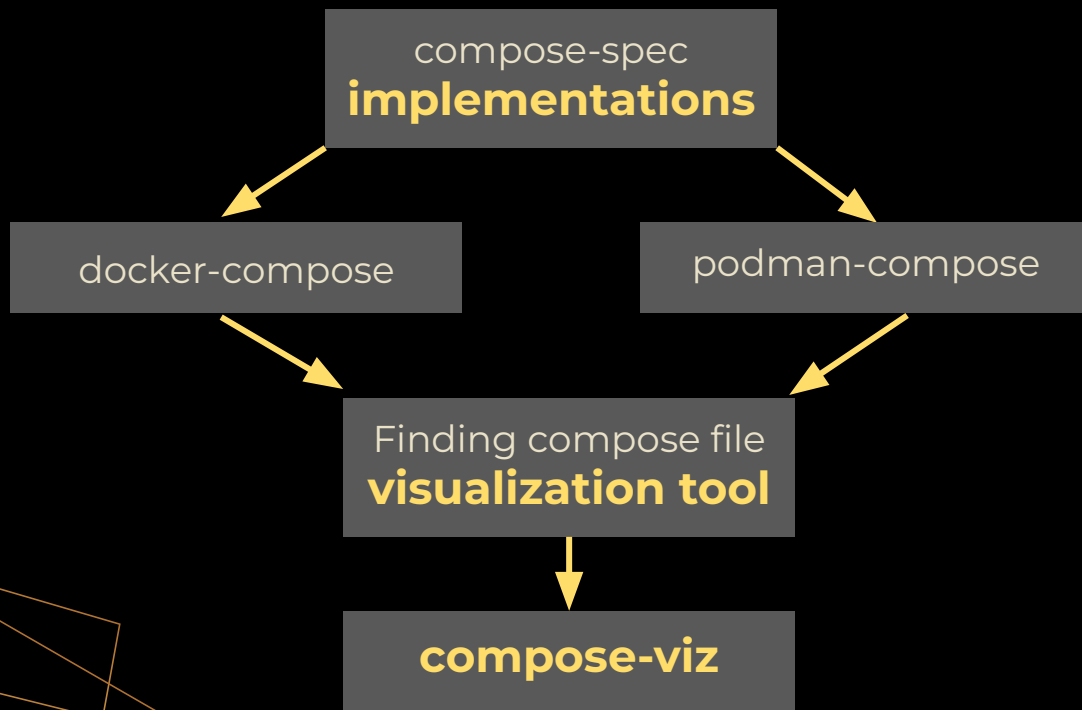
6. Future Work



1

About compose-viz

Motivation





compose-viz/**compose-viz**

A compose file visualization tool that follows compose-spec and allows you to generate graph in several formats.

CONTRIBUTORS

3

FORKS

0

STARS

24

LICENSE

MIT



compose-viz 0.2.3

```
pip install compose-viz
```





2

How it works

Input

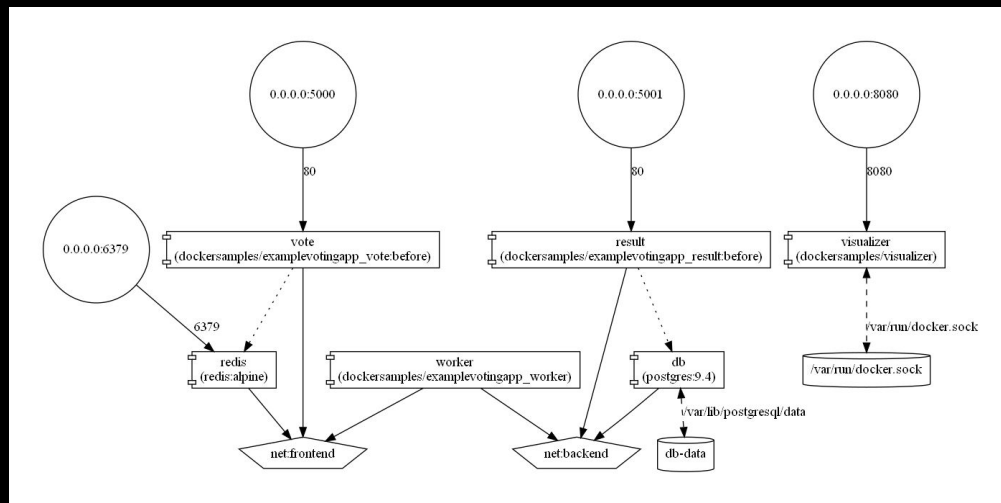
docker-compose.yml

```
version: "3.9"

services:
  redis:
    image: redis:alpine
    ports:
      - "6379"
    networks:
      - frontend
    deploy:
      replicas: 2
      update_config:
        parallelism: 2
      delay: 10s
      restart_policy:
        condition: on-failure
  db:
    image: postgres:9.4
    volumes:
      - db-data:/var/lib/postgresql/data
    networks:
      - backend
    deploy:
      placement:
        constraints: [node.role == manager]
  vote:
    image: dockersamples/examplevotingapp_vote:before
    ports:
      - 5000:80
    networks:
      - frontend
    depends_on:
      - redis
    deploy:
      replicas: 2
      update_config:
        parallelism: 2
      restart_policy:
        condition: on-failure
  result:
    image: dockersamples/examplevotingapp_result:before
    ports:
      - 5001:80
    networks:
      - backend
    depends_on:
      - db
    deploy:
      replicas: 2
      update_config:
        parallelism: 2
      restart_policy:
        condition: on-failure
  worker:
    image: dockersamples/examplevotingapp_worker
    networks:
      - backend
  visualizer:
    image: dockersamples/visualizer
    ports:
      - 8080:8080
    networks:
      - frontend
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
```

Example

Output



compose-viz.png /svg ...
(40 different formats)

Installation

Using `pip`

```
pip install compose-viz
```

Using `.whl`

Updated in [v0.2.3](#)
release

Docker Image

```
docker pull  
wst24365888/compose-viz
```

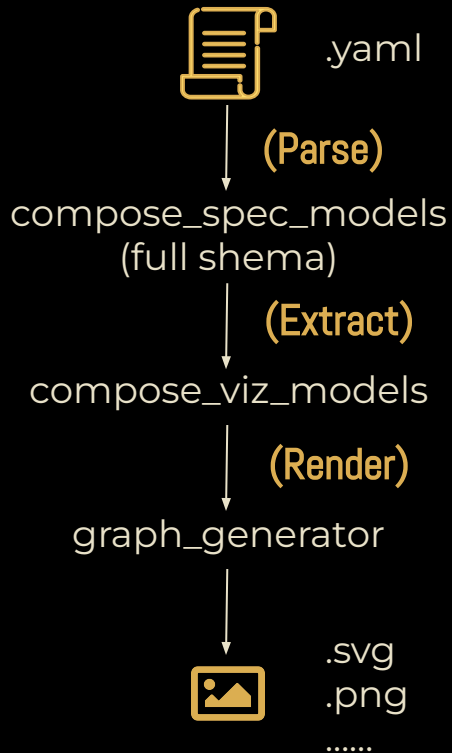
Usage & Options

`cpv [OPTIONS] INPUT_PATH`

Option	Description
<code>-o, --output-filename</code>	Output filename for the generated visualization file. [default: compose-viz]
<code>-m, --format</code>	Output format for the generated visualization file. See supported formats . [default: png]
<code>-v, --version</code>	Show the version of compose-viz.
<code>--help</code>	Show help and exit.

System Structure & Workflow

- cli
 - parse args
 - show help
- parser
 - parse yaml file recursively
- graph generator
 - Graphviz
 - generate multiple formats





3

Testing

43

Test Cases

13

Visualization Components

100%

Code Coverage

67 (422)

Assertions

Testing- Pytest

- Test Cases
 - According to [The Compose Specification](#)
- E2E
 - From cil to graph output
- Integration Test
 - cil -> parser
 - Inside parser
 - Parser output -> graph generator



E2E (test_cli.py)

```
10
11 @pytest.mark.parametrize(
12     "test_file_path",
13     [
14         "tests/ymls/builds/docker-compose.yml",
15         "tests/ymls/cgroup_parent/docker-compose.yml",
16         "tests/ymls/container_name/docker-compose.yml",
17         "tests/ymls/depends_on/docker-compose.yml",
18         "tests/ymls/devices/docker-compose.yml",
19         "tests/ymls/env_file/docker-compose.yml",
20         "tests/ymls/expose/docker-compose.yml",
21         "tests/ymls/extends/docker-compose.yml",
22         "tests/ymls/links/docker-compose.yml",
23         "tests/ymls/networks/docker-compose.yml",
24         "tests/ymls/ports/docker-compose.yml",
25         "tests/ymls/profiles/docker-compose.yml",
26         "tests/ymls/volumes/docker-compose.yml",
27         "examples/full-stack-node-app/docker-compose.yml",
28         "examples/non-normative/docker-compose.yml",
29     ],
30 )
31
32 def test_cli(test_file_path: str) -> None:
33     input_path = f"{test_file_path}"
34     output_filename = "compose-viz-test"
35     default_format = "png"
36     result = runner.invoke(cli.app, ["-o", output_filename, input_path])
37
38     assert result.exit_code == 0
39     assert f"Successfully parsed {input_path}\n" in result.stdout
40     assert os.path.exists(f"{output_filename}.{default_format}")
41
42     os.remove(f"{output_filename}.{default_format}")
```

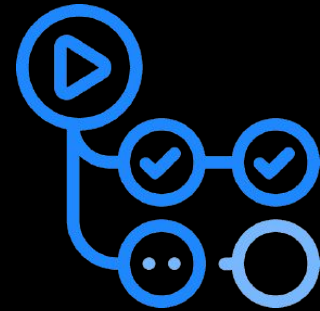
Integration Level (test_parse_file.py)

```
@pytest.mark.parametrize(
    "test_file_path, expected",
    [
        (
            "builds/docker-compose",
            Compose(
                services=[
                    Service(
                        name="frontend",
                        image="build from './frontend', image: awesome/frontend",
                    ),
                    Service(
                        name="backend",
                        image="build from 'backend' using './backend.Dockerfile'",
                    ),
                    Service(
                        name="db",
                        image="build from './db'",
                    ),
                ],
            ),
        ),
    ],
)
```

```
364 def test_parse_file(test_file_path: str, expected: Compose) -> None:
365     parser = Parser()
366     actual = parser.parse(f"tests/ymls/{test_file_path}.yml")
367
368     assert len(actual.services) == len(expected.services)
369
370     for actual_service, expected_service in zip(actual.services, expected.services):
371         assert actual_service.name == expected_service.name
372         assert actual_service.image == expected_service.image
373
374         assert len(actual_service.ports) == len(expected_service.ports)
375         for actual_port, expected_port in zip(actual_service.ports, expected_service.ports):
376             assert actual_port.host_port == expected_port.host_port
377             assert actual_port.container_port == expected_port.container_port
378             assert actual_port.protocol == expected_port.protocol
379
380         assert actual_service.networks == expected_service.networks
381
382         assert len(actual_service.volumes) == len(expected_service.volumes)
383         for actual_volume, expected_volume in zip(actual_service.volumes, expected_service.volumes):
384             assert actual_volume.source == expected_volume.source
385             assert actual_volume.target == expected_volume.target
386             assert actual_volume.type == expected_volume.type
387
388         assert actual_service.depends_on == expected_service.depends_on
389         assert actual_service.links == expected_service.links
390
391         assert (actual_service.extends is not None) == (expected_service.extends is not None)
392
393         if (actual_service.extends is not None) and (expected_service.extends is not None):
394             assert actual_service.extends.service_name == expected_service.extends.service_name
395             assert actual_service.extends.from_file == expected_service.extends.from_file
396
397         assert actual_service.cgroup_parent == expected_service.cgroup_parent
398         assert actual_service.container_name == expected_service.container_name
399
400         assert actual_service.expose == expected_service.expose
401         assert actual_service.env_file == expected_service.env_file
402         assert actual_service.profiles == expected_service.profiles
403
404         assert len(actual_service.devices) == len(expected_service.devices)
405         for actual_device, expected_device in zip(actual_service.devices, expected_service.devices):
406             assert actual_device.host_path == expected_device.host_path
407             assert actual_device.container_path == expected_device.container_path
408             assert actual_device.cgroup_permissions == expected_device.cgroup_permissions
409
```


Testing- Continuous Integration

- Github Actions
 - Test
 - When push or PR on branch main & dev
 - Github Release
 - On every push of tag: v*
 - Publish to PyPi
 - On every push of tag: v*



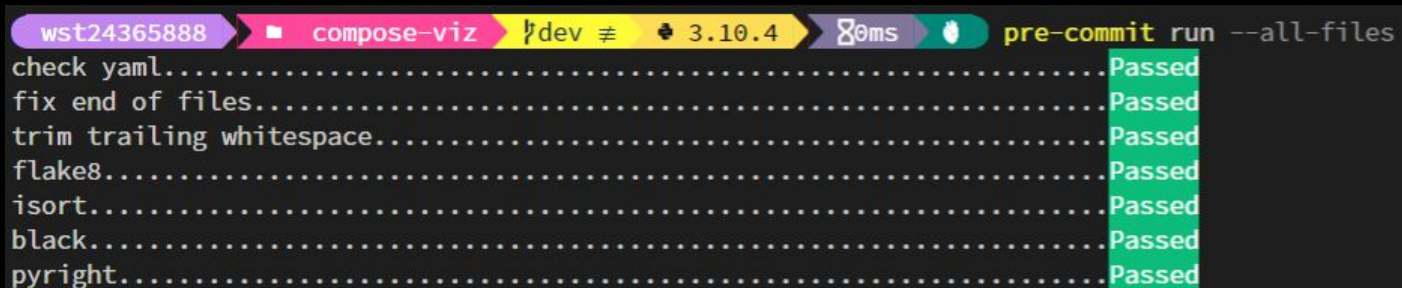


4

Pre-commit

Pre-commit

- pre-commit hooks
 - flake8: linter
 - isort: sort import dependencies
 - black: formatter
 - pyright: static type checker
- Ensure code quality before each commit.



A terminal window showing the execution of pre-commit hooks. The window title bar includes the username 'wst24365888', the project name 'compose-viz', the branch 'dev', the Python version '3.10.4', and the editor '8ms'. The command 'pre-commit run --all-files' has been executed. The output shows seven checks, all of which passed:

Check	Result
check yamll.....	Passed
fix end of files.....	Passed
trim trailing whitespace.....	Passed
flake8.....	Passed
isort.....	Passed
black.....	Passed
pyright.....	Passed



5

Experience Sharing



Things To Share

- the test progressed too fast at the beginning, which was a bit frustrating when developing
- need to think more when implementing
- static type checking is very helpful
- very useful when refactoring



6

Future Work

Roadmap



- ❑ Support more visualization components
- ❑ Add more tests, like mutation tests

See [open issues](#) for a full list of proposed features (and known issues).

THANKS!

Q & A

Software Testing Final Project

Team 14



<https://github.com/compose-viz/compose-viz>