Pytest Plugins

как расширить функционал тестового фреймворка через плагины

Александр Волков

Тимлид AQA-команды, YADRO

Более 6 лет в Automation QA. Писал тесты на банковское ПО, распределенные базы данных, на сервера и системы хранения данных. Участвовал в разработке нескольких фреймворков для автоматизации тестирования.

Кто мы?

- R&D в Москве, Санкт-Петербурге, Нижнем Новгороде и Минске
- Собственная производственная площадка в МО.
- 3500+ сотрудников
- 80% сотрудников – инженерные команды

Что мы делаем?

Высокопроизводительные серверы

- VEGMAN

Системы хранения данных (СХД)

- TATLIN.ARCHIVE
- TATLIN.UNIFIED
- TATLIN.OBJECT
- TATLIN.FLEX

Направления тестирования

- Использование системы потребителем (Data path).
- Менеджмент системы (Control path).
- Совместимость с конечными устройствами (различные OS, системы виртуализации).
- Тестирование интеграции с другими системами (системы мониторинга, аутентификации).

Короткий обзор pytest

Pytest hooks

Pytest plugins

Примеры плагинов из нашего фреймворка

Plugin pitfalls

~~Короткий обзор pytest~~

Pytest hooks

Pytest plugins

Примеры плагинов из нашего фреймворка

Plugin pitfalls

Pytest

https://github.com/pytest-dev/pytest (> 10 k stars)

https://docs.pytest.org/en/6.2.x/contents.html

Pytest features:

- Detailed info on failing assert statements.
- Auto-discovery of test modules and functions.
- Modular fixtures for managing small or parametrized long-lived test resources.
- Rich plugin architecture, with over 1316+ external plugins and thriving community.

Pytest
(жизненный цикл запуска тестов)

- Создание внутренних объектов pytest (Config, Session)
- Регистрация плагинов

- Tests Discovery
- Фильтрация тестов

Запуск тестов:
Выполнение
run test protocol
для каждого теста

Формирование отчета

Pytest
(нестандартные задачи)

- Передать опции командной строки, меняющие поведение тестов (конфигурационный файл).
- Отфильтровать тесты, которые не подходят по конфигурации (версия продукта).
- Положить результат каждого теста в BD сразу после выполнения теста.
- Сформировать отчет, который можно загрузить в вашу систему для отчетов.

Короткий обзор pytest

Pytest hooks

Pytest plugins

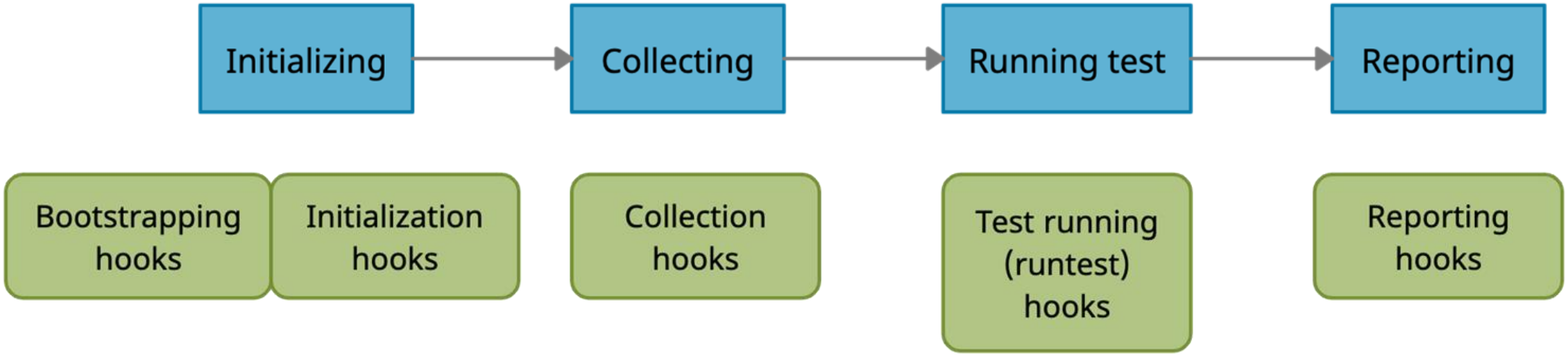
Примеры плагинов из нашего фреймворка

Plugin pitfalls

Pytest hooks

- Функции с определенным именем (начинается с pytest_) и сигнатурой.
- Реализованы в conftest.py файле или в плагине.
- https://docs.pytest.org/en/6.2.x/reference.html#hooks
- Вызываются самим pytest в определенные моменты.

Pytest hooks

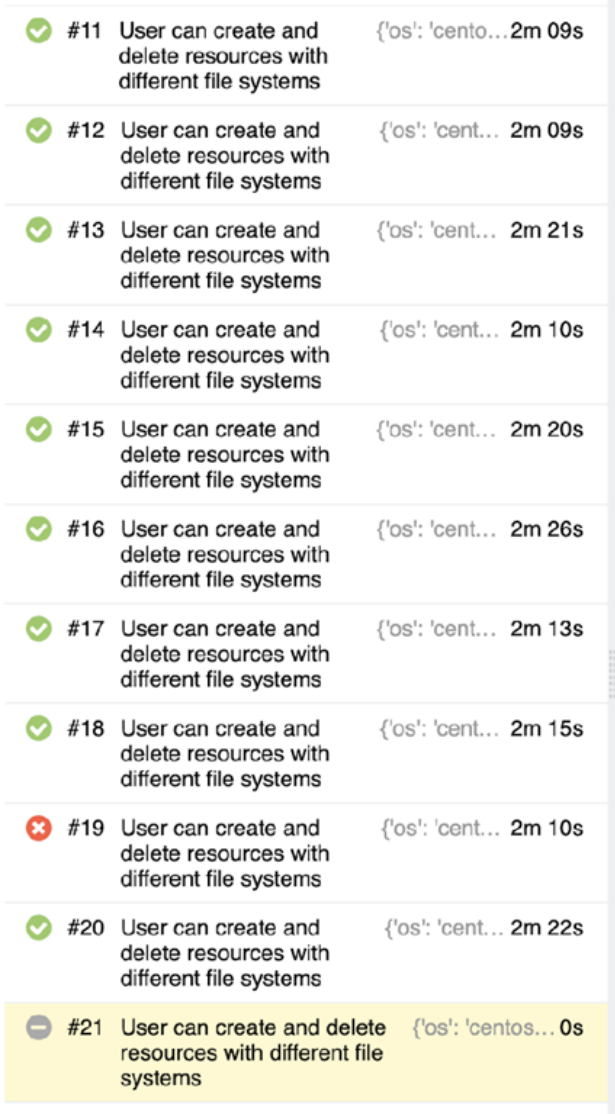


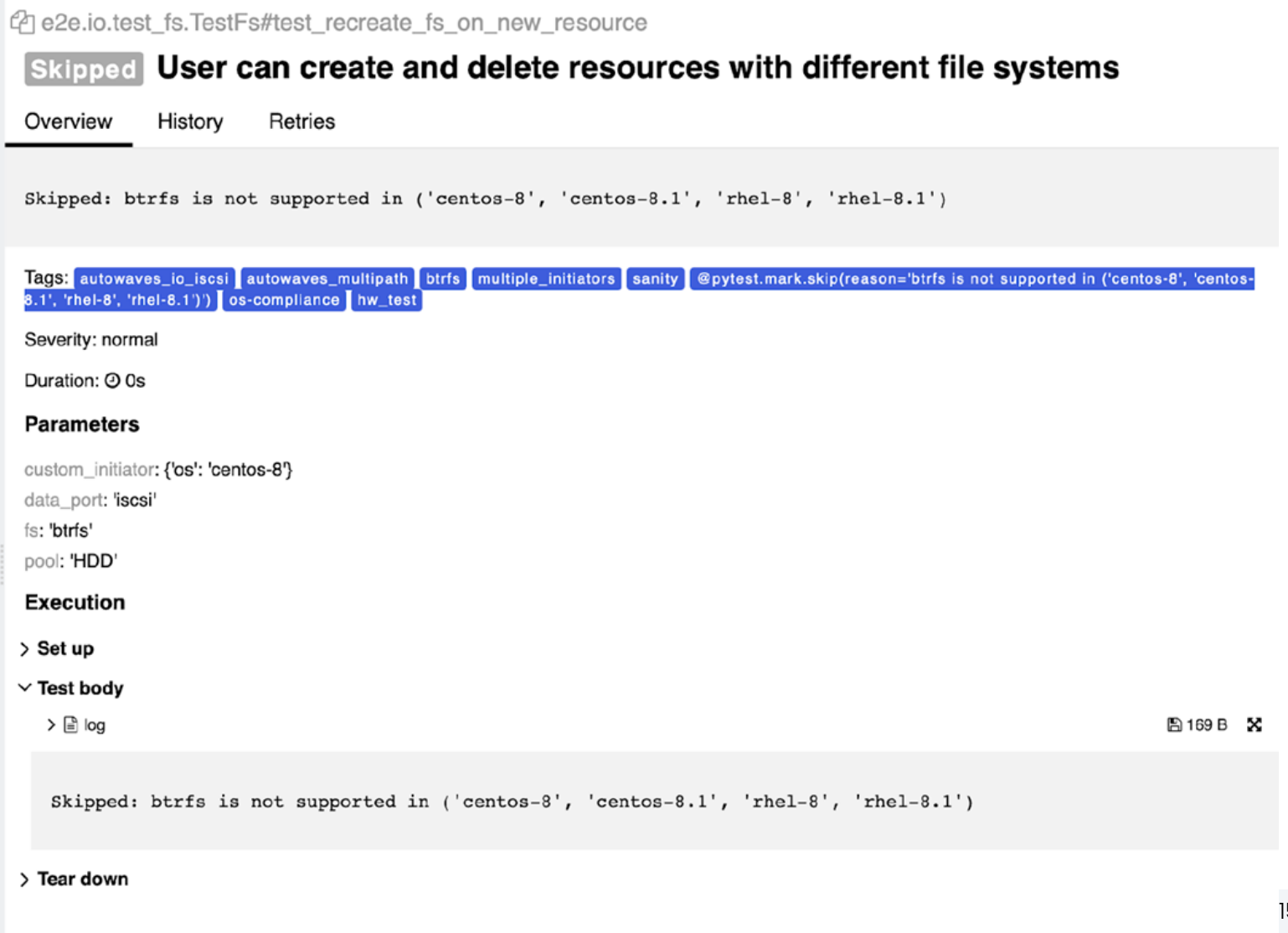
Примеры использования hook-ов из нашего проекта

- Автоматически исключать из выполнения тесты, не подходящие по конфигурации.

```
def pytest_collection_modifyitems(items: List[Item]):
    not_supported_os = 'centos-8', 'centos-8.1', 'rhel-8', 'rhel-8.1'
    for test_case in items:
        if 'test_fs' in test_case.nodeid:
            initiator_info = test_case.callspec.params.get('custom_initiator', {})
            fs = test_case.callspec.params.get('fs')
            if fs == 'btrfs' and initiator_info.get('os') in not_supported_os:
                test_case.add_marker(pytest.mark.skip(reason=f'btrfs is not supported in {not_supported_os}'))
```

Примеры использования hook-ов из нашего проекта





Примеры использования hook-ов из нашего проекта

- Добавляем для каждого теста в Allure отчет ссылку на логи

```
395  def pytest_runtest_setup(item: Item):
396      job_name = os.getenv('SYS_TESTS_JOB_NAME')
397      wave_name = os.getenv('WAVE_NAME')
398      wave_id = os.getenv('BUILD_ID')
399      if job_name and wave_name:
400          logs_link = f'http://logs.spb.yadro.com/ci-logs/{job_name}/artifacts/logs/{wave_name}/'
401          allure.dynamic.link(logs_link, name='LOGS')
```

- e2e.network 18
- e2e.notification 45 5
- e2e.pools 111
- e2e.recovery 6
- e2e.resources 121 1
- e2e.rest 1
- e2e.snmp 5
- e2e.stat 10 1
- e2e.system 30 2
- e2e.tatlin_reports 2
- e2e.time 7
- e2e.ui 21 1
  - Total: **22m 01s** Max: **18s 864ms** Sum: **2m 09s**
  - test_provision 16 1
  - test_set_user_role 5
    - Total: **2m 40s** Max: **14s 001ms** Sum: **47s 580ms**
    - TestUserSettingsUI 5
      - Total: **2m 40s** Max: **14s 001ms** Sum: **47s 580ms**
      - #1 Admin user can create new user 14s 001ms
      - #2 Newly created user is able to login 3s 403ms
      - #3 Admin user can change user role 13s 819ms
      - #4 Admin user can delete user 13s 006ms
      - #5 Deleted user is not able to login 3s 351ms

Links

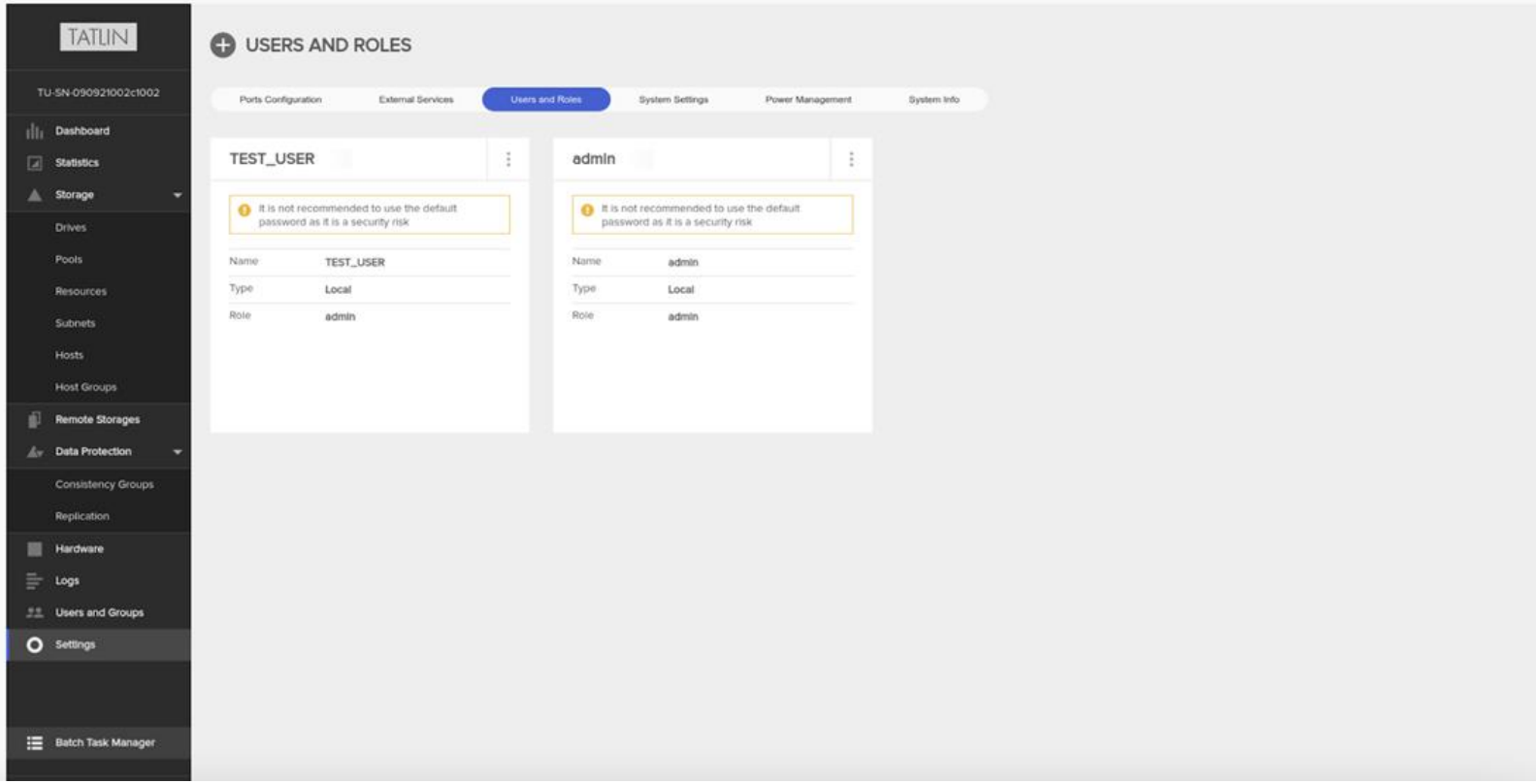
LOGS , ALLURE WAVE REPORT

Execution

Set up

Test body

- Login to tatlin 1 parameter, 11 sub-steps 3s 692ms
- Create user 4 parameters, 30 sub-steps 6s 641ms
- Get all users 2 parameters, 6 sub-steps 2s 500ms
- Check in CLI created user 4 sub-steps, 1 attachment 1s 166ms
- screenshot after test - test_add_new_user 76.2 KB



Pytest hooks

- Вызываются самим pytest в определенные моменты.
- https://docs.pytest.org/en/7.1.x/reference/reference.html?highlight=pytest_runtest_protocol#hooks - описаны все hook функции, нужно только выбрать нужную.
- https://github.com/pytest-dev/pytest/issues/3261#issuecomment-1670996125 - веселые картинки

Короткий обзор pytest

Pytest hooks

~~Pytest plugins~~

Примеры плагинов из нашего фреймворка

Plugin pitfalls

Pytest plugins

Плагин – это код, который:

- Содержит одну или несколько hook функций.
- Решает одну конкретную задачу (Single Responsibility).
- Не связан с тестами.

Pytest plugins

Примеры популярных плагинов, которые мы используем:

- allure-pytest - репортинг результатов тестов через allure-framework.

  https://github.com/allure-framework/allure-python

- pytest-timeout - позволяет выставлять таймаут для тестов глобально или для каждого конкретного теста через марки.

  https://github.com/pytest-dev/pytest-timeout

- pytest-xdist - позволяет “параллелить” прохождение тестов.

  https://github.com/pytest-dev/pytest-xdist

- pytest-stress - позволяет запускать тесты “по кругу” в течении заданного периода времени.

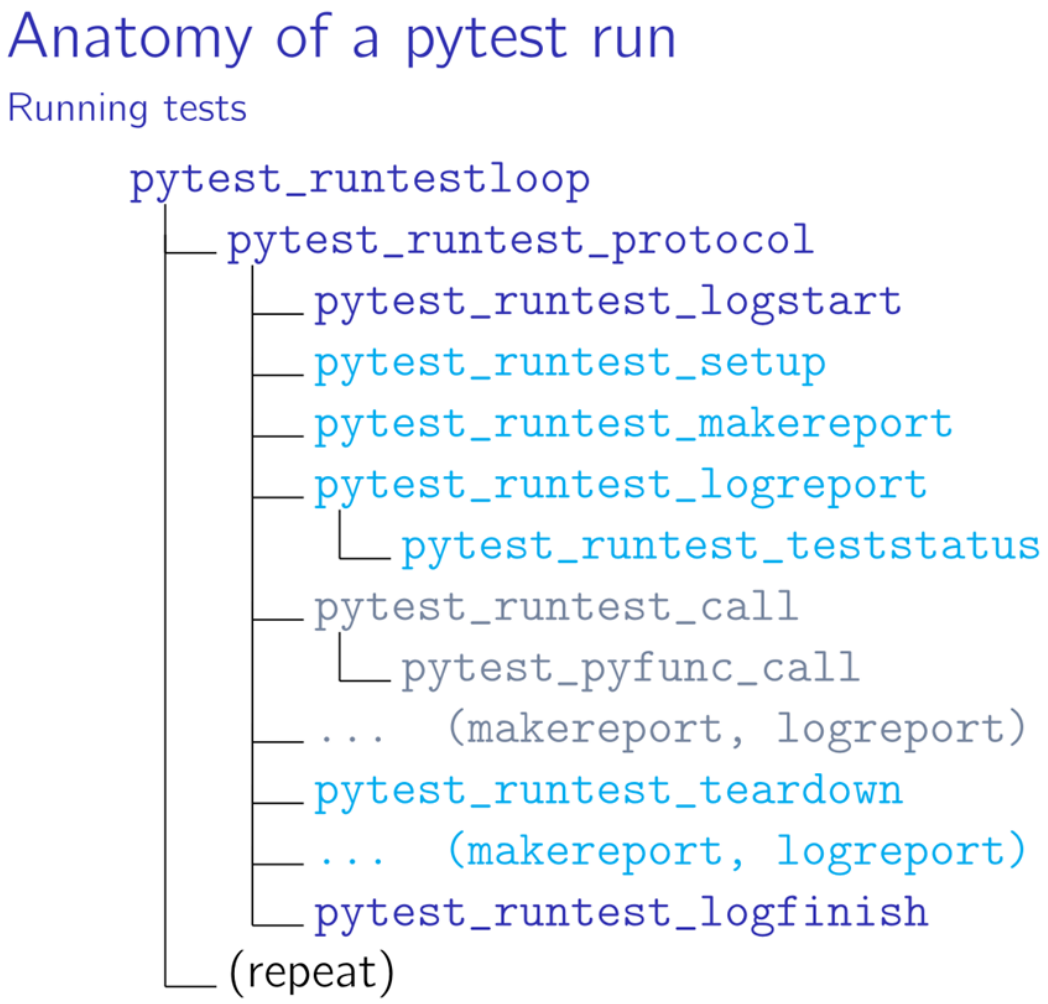
  https://github.com/ImXron/pytest-stress

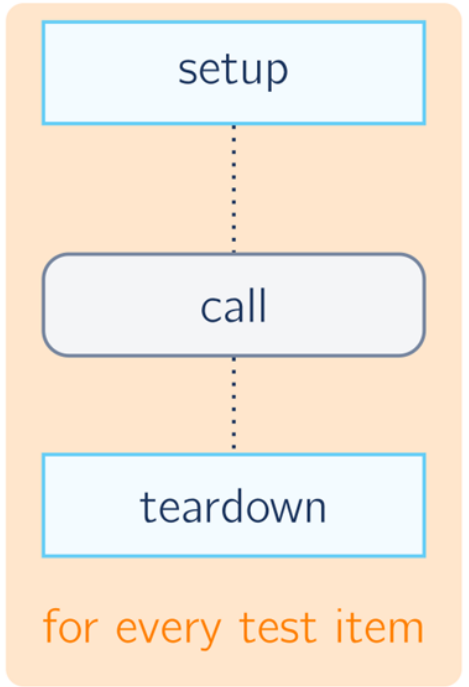
Pytest plugins: pytest-stress

Плагин – это код, который:

Содержит одну или несколько hook функций.

Pytest plugins: pytest-stress



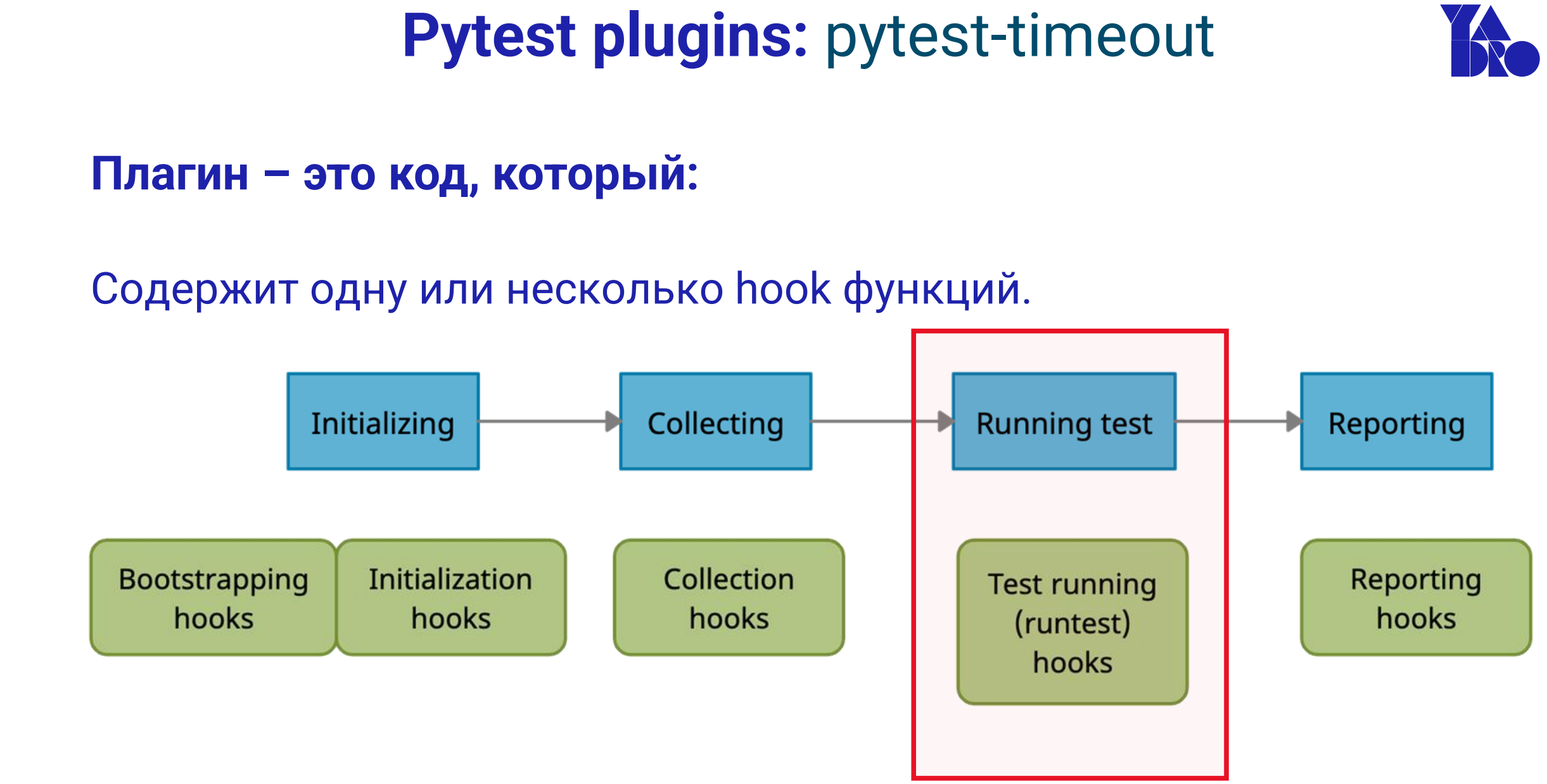


https://github.com/pytest-dev/pytest/issues/3261#issuecomment-1670996125

Pytest plugins: pytest-stress

```
97  def pytest_runtestloop(session):
98      """Reimplement the test loop but loop for the user defined amount of time...."""
103     ...
111
112     while True:
113         if _get_total_time(session):
114             _print_loop_count(count)
115         for index, item in enumerate(session.items):
116             next_item = session.items[index + 1] if index + 1 < len(session.items) else None
117             item.config.hook.pytest_runtest_protocol(item=item, nextitem=next_item)
118             if session.shouldfail:
119                 raise session.Failed(session.shouldfail)
120             if session.shouldstop:
121                 raise session.Interrupted(session.shouldstop)
122         count += 1
123         if _timed_out(session, start_time):
124             break
125         time.sleep(_get_delay_time(session))
126     return True
```

Pytest plugins: pytest-timeout



Плагин – это код, который:

Содержит одну или несколько hook функций.

Pytest plugins: pytest-timeout

```
148  @pytest.hookimpl(hookwrapper=True)
149  def pytest_runtest_protocol(item):
150      """Hook in timeouts to the runtest protocol...."""
156      hooks = item.config.pluginmanager.hook
157      settings = _get_item_settings(item)
158      is_timeout = settings.timeout is not None and settings.timeout > 0
159      if is_timeout and settings.func_only is False:
160          hooks.pytest_timeout_set_timer(item=item, settings=settings)
161      yield
162      if is_timeout and settings.func_only is False:
163          hooks.pytest_timeout_cancel_timer(item=item)
```

Короткий обзор pytest

Pytest hooks

Pytest plugins

Примеры плагинов из нашего фреймворка

Plugin pitfalls

pytest-affected

Проблема:

- При открытии PR-а в репозиторий тестов запускается **большая** часть тестов.
- **Неэффективно** тратятся ресурсы CI.
- Приходится **долго** ждать чтобы замержить изменения в мастер.

pytest-affected

Идея:

Запускать только те тесты, которые покрывают измененный код.

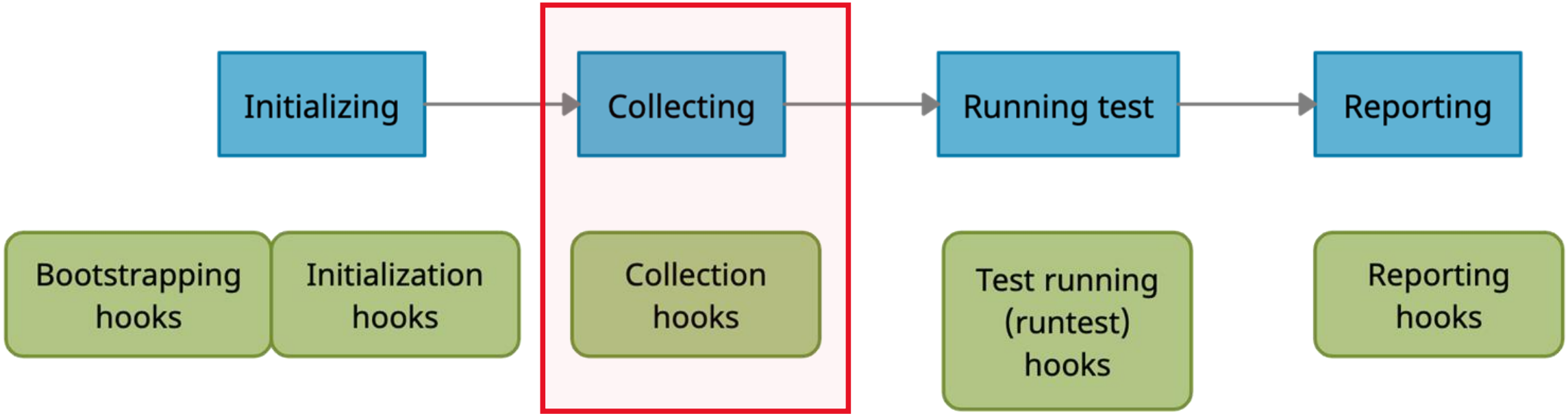
Реализация:

- Взять с помощью git изменения.
- Определить какие тесты покрывают эти изменения.
- Запустить эти тесты.
- Реализовать это в виде плагина.

pytest-affected

Плагин – это код, который:

Содержит одну или несколько hook функций.



pytest-affected

```
33  def pytest_collection_modifyitems(session, config, items):
34      ...
40      changes = get_changes_from_git(
41          config.getoption("git_path"), config.getoption("git_branch")
42      )
43      if not changes:...
47      rules = get_rules(config.getoption("affected_rules"))
48      test_filenames, updated_changes = process_python_files(
49          config.getoption("git_path"), rules, changes
50      )
51      if updated_changes:
52          test_filenames.update(
53              process_not_python_files(
54                  config.getoption("git_path"), rules, updated_changes
55              )
56          )
57          test_filenames.update(
58              get_affected_test_filenames(config.getoption("project"), updated_changes)
59          )
```

pytest-affected

```
67      selected = []
68      deselected = []
69      for item in items:
70          item_path = item.location[0]
71          if any(item_path in test_filename for test_filename in test_filenames):
72              selected.append(item)
73              continue
74          deselected.append(item)
75      items[:] = selected
76      if deselected:
77          config.hook.pytest_deselected(items=deselected)
```

pytest-affected

PROFIT!!!

Что получаем:

Best case: тесты не запускаются/запускаются только нужные.

Average case: запускается только малая часть тестов.

Worst case: запускаются все тесты.

Короткий обзор pytest

Pytest hooks

Pytest plugins

Примеры плагинов из нашего фреймворка

~~Plugin pitfalls~~

Что происходит, если несколько плагинов в одном проекте используют одинаковые hook – функции?

Плагины используют одинаковые hook – функции

```python
# Plugin 1
@pytest.hookimpl(tryfirst=True)
def pytest_collection_modifyitems(items):
    # will execute as early as possible
    ...


# Plugin 2
@pytest.hookimpl(trylast=True)
def pytest_collection_modifyitems(items):
    # will execute as late as possible
    ...


# Plugin 3
@pytest.hookimpl(hookwrapper=True)
def pytest_collection_modifyitems(items):
    # will execute even before the tryfirst one above!
    outcome = yield
    # will execute after all non-hookwrappers executed
```

Плагины используют одинаковые hook – функции

```
18  # Plugin 1
19  @pytest.hookimpl(tryfirst=True)
20  def pytest_collection_modifyitems(items):
21      # will execute as early as possible
22      ...
23
24
25  # Plugin 2
26  @pytest.hookimpl(trylast=True)
27  def pytest_collection_modifyitems(items):
28      # will execute as late as possible
29      ...
30
31
32  # Plugin 3
33  @pytest.hookimpl(hookwrapper=True)
34  def pytest_collection_modifyitems(items):
35      # will execute even before the tryfirst one above!
36      outcome = yield
37      # will execute after all non-hookwrappers executed
```

2

3

1

4

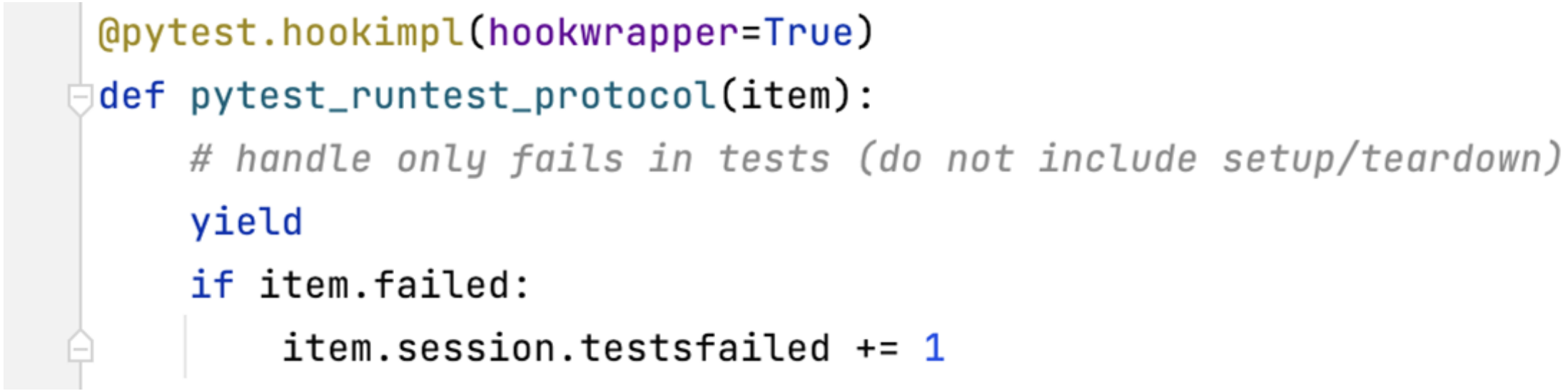
История одного traceback-а

```
[2023-08-13T22:59:44.492Z] INTERNALERROR> Traceback (most recent call last):
[2023-08-13T22:59:44.492Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/_pytest/main.py", line 270, in wrap_session
[2023-08-13T22:59:44.493Z] INTERNALERROR>     session.exitstatus = doit(config, session) or 0
[2023-08-13T22:59:44.493Z] INTERNALERROR>                          ^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/_pytest/main.py", line 324, in _main
[2023-08-13T22:59:44.493Z] INTERNALERROR>     config.hook.pytest_runtestloop(session=session)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_hooks.py", line 433, in __call__
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._hookexec(self.name, self._hookimpls, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_manager.py", line 112, in _hookexec
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._inner_hookexec(hook_name, methods, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_callers.py", line 155, in _multicall
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return outcome.get_result()
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_result.py", line 108, in get_result
[2023-08-13T22:59:44.493Z] INTERNALERROR>     raise exc.with_traceback(exc.__traceback__)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_callers.py", line 80, in _multicall
[2023-08-13T22:59:44.493Z] INTERNALERROR>     res = hook_impl.function(*args)
[2023-08-13T22:59:44.493Z] INTERNALERROR>           ^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pytest_stress/pytest_stress.py", line 117, in pytest_runtestloop
[2023-08-13T22:59:44.493Z] INTERNALERROR>     item.config.hook.pytest_runtest_protocol(item=item, nextitem=next_item)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_hooks.py", line 433, in __call__
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._hookexec(self.name, self._hookimpls, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_manager.py", line 112, in _hookexec
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._inner_hookexec(hook_name, methods, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_callers.py", line 133, in _multicall
[2023-08-13T22:59:44.493Z] INTERNALERROR>     teardown[0].send(outcome)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/root/suites/conftest.py", line 299, in pytest_runtest_protocol
[2023-08-13T22:59:44.493Z] INTERNALERROR>     if item.failed:
[2023-08-13T22:59:44.493Z] INTERNALERROR>        ^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR> AttributeError: 'Function' object has no attribute 'failed'
[2023-08-13T22:59:44.751Z] ---------------------------- live log sessionfinish ----------------------------
```

История одного traceback-a

```
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pytest_stress/pytest_stress.py", line 117, in pytest_runtestloop
[2023-08-13T22:59:44.493Z] INTERNALERROR>     item.config.hook.pytest_runtest_protocol(item=item, nextitem=next_item)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_hooks.py", line 433, in __call__
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._hookexec(self.name, self._hookimpls, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_manager.py", line 112, in _hookexec
[2023-08-13T22:59:44.493Z] INTERNALERROR>     return self._inner_hookexec(hook_name, methods, kwargs, firstresult)
[2023-08-13T22:59:44.493Z] INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/usr/local/lib/python3.11/site-packages/pluggy/_callers.py", line 133, in _multicall
[2023-08-13T22:59:44.493Z] INTERNALERROR>     teardown[0].send(outcome)
[2023-08-13T22:59:44.493Z] INTERNALERROR>   File "/root/suites/conftest.py", line 299, in pytest_runtest_protocol
[2023-08-13T22:59:44.493Z] INTERNALERROR>     if item.failed:
[2023-08-13T22:59:44.493Z] INTERNALERROR>        ^^^^^^^^^^^
[2023-08-13T22:59:44.493Z] INTERNALERROR> AttributeError: 'Function' object has no attribute 'failed'
[2023-08-13T22:59:44.751Z] ---------------------------- live log sessionfinish ----------------------------
```

История одного traceback-а



```python
@pytest.hookimpl(hookwrapper=True)
def pytest_runtest_protocol(item):
    # handle only fails in tests (do not include setup/teardown)
    yield
    if item.failed:
        item.session.testsfailed += 1
```

```
[2023-08-13T22:59:44.493Z] INTERNALERROR>
[2023-08-13T22:59:44.493Z] INTERNALERROR>
[2023-08-13T22:59:44.493Z] INTERNALERROR>
[2023-08-13T22:59:44.493Z] INTERNALERROR>
```

File "/root/suites/conftest.py", line 299, in pytest_runtest_protocol
  if item.failed:
     ^^^^^^^^^^

AttributeError: 'Function' object has no attribute 'failed'

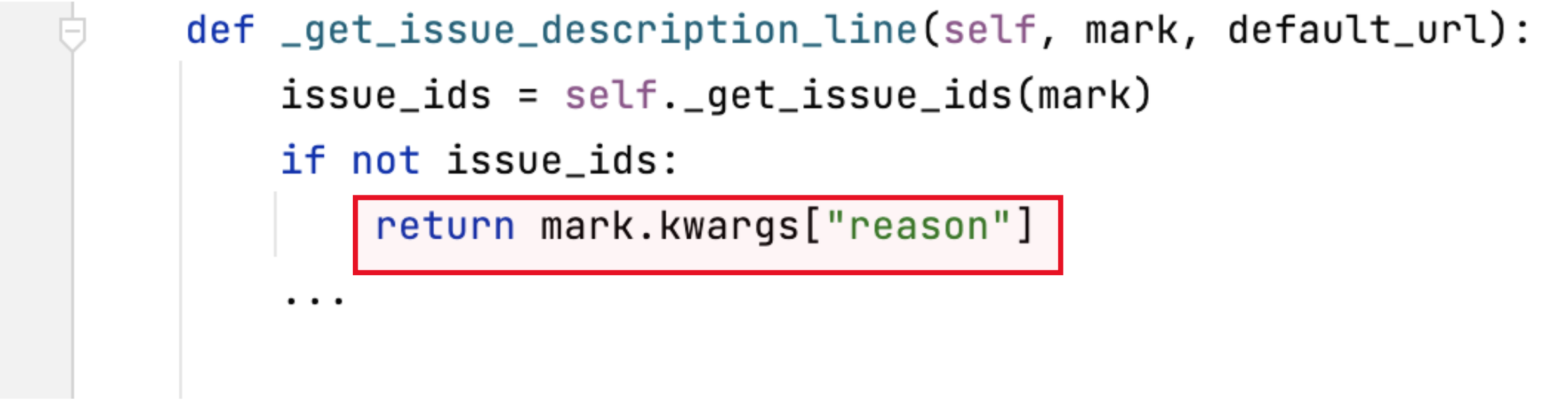
История одного traceback-а

```python
@pytest.hookimpl(hookwrapper=True)
def pytest_runtest_protocol(item):
    # handle only fails in tests (do not include setup/teardown)
    yield
    # if item.failed:
    if hasattr(item, 'failed'):
        item.session.testsfailed += 1
```

История одного traceback-а

```
...
INTERNALERROR>     self._process_metadata_item_start(current_leaf)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-
packages/pytest_reportportal/service.py", line 643, in _process_metadata_item_start
INTERNALERROR>     leaf['issue'] = self._process_issue(item)
INTERNALERROR>                     ^^^^^^^^^^^^^^^^^^^^^^^^^
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-
packages/pytest_reportportal/service.py", line 606, in _process_issue
INTERNALERROR>     return self._get_issue(issues[0])
INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-
packages/pytest_reportportal/service.py", line 542, in _get_issue
INTERNALERROR>     self._get_issue_description_line(mark, default_url)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-
packages/pytest_reportportal/service.py", line 520, in _get_issue_description_line
INTERNALERROR>     return mark.kwargs["reason"]
INTERNALERROR>            ~~~~~~~~~~~~~~^^^^^^^^^^
INTERNALERROR> KeyError: 'reason'
```

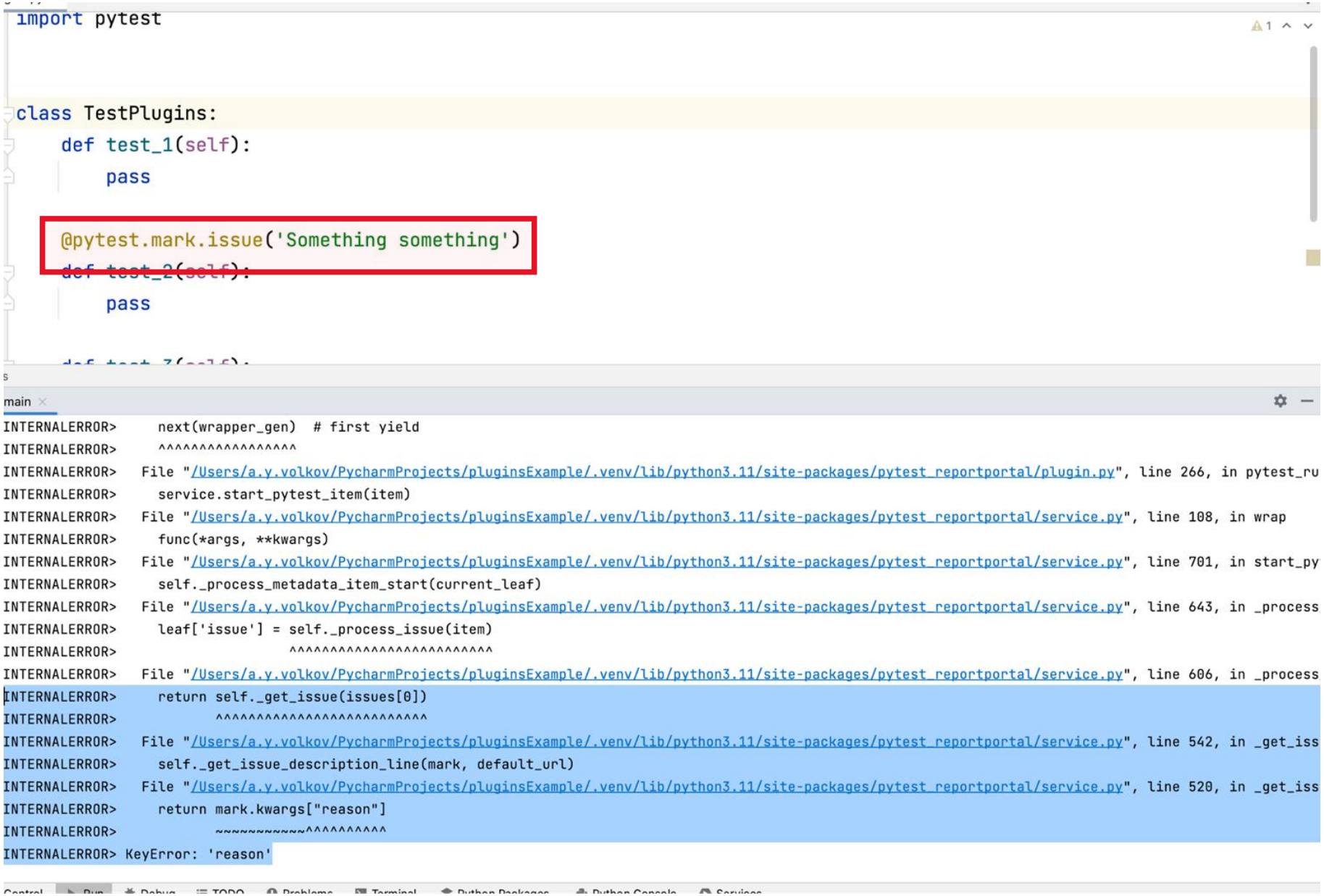
История одного traceback-а



```
def _get_issue_description_line(self, mark, default_url):
    issue_ids = self._get_issue_ids(mark)
    if not issue_ids:
        return mark.kwargs["reason"]
    ...
```

INTERNALERROR> self._get_issue_description_line(mark, default_url)
INTERNALERROR> **File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 520, in _get_issue_description_line**
INTERNALERROR> **return mark.kwargs["reason"]**
INTERNALERROR> **~~~~~~~~~~~~^^^^^^^^^^**
INTERNALERROR> **KeyError: 'reason'**

История одного traceback-а



```python
import pytest


class TestPlugins:
    def test_1(self):
        pass

    @pytest.mark.issue('Something something')
    def test_2(self):
        pass
```

```
main
INTERNALERROR>     next(wrapper_gen)  # first yield
INTERNALERROR>     ^^^^^^^^^^^^^^^^
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/plugin.py", line 266, in pytest_ru
INTERNALERROR>     service.start_pytest_item(item)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 108, in wrap
INTERNALERROR>     func(*args, **kwargs)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 701, in start_py
INTERNALERROR>     self._process_metadata_item_start(current_leaf)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 643, in _process
INTERNALERROR>     leaf['issue'] = self._process_issue(item)
INTERNALERROR>                     ^^^^^^^^^^^^^^^^^^^^^^^^^
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 606, in _process
INTERNALERROR>     return self._get_issue(issues[0])
INTERNALERROR>            ^^^^^^^^^^^^^^^^^^^^^^^^^^
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 542, in _get_iss
INTERNALERROR>     self._get_issue_description_line(mark, default_url)
INTERNALERROR>   File "/Users/a.y.volkov/PycharmProjects/pluginsExample/.venv/lib/python3.11/site-packages/pytest_reportportal/service.py", line 520, in _get_iss
INTERNALERROR>     return mark.kwargs["reason"]
INTERNALERROR>            ~~~~~~~~~~~^^^^^^^^^^
INTERNALERROR> KeyError: 'reason'
```

Кто виноват? Что делать?

- Внимательно смотрите какие pytest plugins подгружаются при прогоне тестов.
- Заглядывайте в плагины, которые вы используете, чтобы понимать как они работают.
- Точкой входа в плагин являются hook функции.
- Не допускайте исключений в реализации hook функций.

О чем поговорили

- Pytest предоставляет несколько возможностей для расширения и изменения поведения фреймворка - pytest hooks и pytest plugins.
- Pytest Plugins – это возможность создавать «кусочки» законченного функционала, меняющие поведение вашего фреймворка.
- Писать плагины к pytest не очень сложно. Можно брать за основу open source плагины.
- Читайте документацию к pytest, она действительно хорошая.
- Не забывайте писать тесты для ваших плагинов.

Спасибо за внимание
Готов ответить на вопросы