







pytest_*

-  устарело
-  только для установки
-  есть несовместимость
-  прочие опасности

-  используется в OneLine
-  может пригодиться

Bootstrapping

- load_initial_conftests**
Загрузка конфигов до разбора параметров командной строки
- cmdline_preparse**
(устарело) Модификация параметров командной строки до её разбора
- cmdline_parse**
Разбирает аргументы строки и возвращает конфигурацию запуска
- cmdline_main**
Основное действие командной строки при установке системы

Initialization

- addoption**
Добавляет опцию командной строки или ini-файла в список распознаваемых
- addhooks**
Добавляет кастомный хук
- configure**
Настраиает конфигурацию сразу после разбора командной строки, и потом каждый раз для очередного conftest.py
- unconfigure**
Перед окончанием процесса тестирования
- sessionstart**
Как только сессия настроена, до запуска сборки и тестирования
- sessionfinish**
Перед окончанием сессии. Это последнее действие перед тем, как вернуть управление системе
- plugin_registered**
Сразу после регистрации нового плагина

Collection

- collection**
Реализует весь процесс сборки тест-кейсов. Хук запускается в начале сборки. Например, используется для вывода количества найденных кейсов?
- ignore_collect**
До запуска остальных хуков по тест-кейсу сообщает, нужно ли вообще обрабатывать этот тест-кейс — True/False
- collect_file**
Создаёт объект Collector для конкретного пути
- pycollect_makemodule**
Создаёт Module collector для каждого тестируемого модуля
- pycollect_makeitem**
Создаёт item/collector для Python-объекта в модуле. Проще говоря, создаёт тест-кейс
- generate_tests**
Параметризует вызовы функции
- make_parametrize_id**
Возвращает строку для параметризации
- collection_modifyitems**
После сборки всех тест-кейсов. Может отфильтровать, отсортировать или иначе изменить весь набор
- collection_finish**
В самом конце этапа сборки тест-кейсов

Test Running

- runtestloop**
Реализует основной цикл тестирования. Запускает протокол тестирования для всех собранных тест-кейсов.
- runtest_protocol**
Реализует протокол тестирования для одного тест-кейса.
- runtest_logstart**
В самом начале протокола тестирования
- runtest_logfinish**
В самом конце протокола тестирования
- runtest_setup**
Реализует фазу настройки конкретного тест-кейса (и его ещё не настроенных предков), в том числе получение его фикстур
- runtest_teardown**
Реализует фазу завершения конкретного тест-кейса (и его предков, которых уже можно завершать), в том числе завершения работы фикстур, которые уже отработали в своём контексте
- runtest_makereport**
Вызывает `_pytest.reports.TestReport` для фаз setup, call и teardown каждого тест-кейса
- pyfunc_call**
Вызывает тестируемую функцию

Reporting

- collectstart**
Collector начинает сборку
- make_collect_report**
Исполняет `collector.collect()` и возвращает `CollectReport`
- itemcollected**
Как только получен (найден, собран) тест-кейс
- collectreport**
В конце работы Collector
- deselected**
При отмене выбора тест-кейса (в том числе по ключевым словам)
- report_header**
Возвращает строку или список строк, которые будут отображаться в терминале как заголовок отчёта о тестировании
- report_collectionfinish**
Возвращает строку или список строк, которые будут отображаться в терминале после успешного завершения отчёта. Эти строки отображаются после стандартного сообщения “collected X items”
- report_teststatus**
Возвращает короткие и расширенные символы для отображения конкретного статуса результата тестирования.

Возможные категории, к которым могут быть привязаны статусы: “passed”, “skipped”, “error” или пустая строка. Возможно, что-то ещё.

Короткая версия — маркер в строке прогресса (например, точка).

Длинная версия — слово (например, “PASSED”).

Можно указывать цвета: “rerun”, “R”, (“RERUN”, {“yellow”:True}).
- terminal_summary**
Добавить раздел в summary отчёта
- fixture_setup**
Запуск при настройке фикстуры
- fixture_post_finalizer**
Запуск после «гашения» (остановки, удаления) фикстуры, но до очистки кэша. Поэтому здесь доступны значения фикстуры: `fixturedef.cached_result`
- warning_captured**
(устарело) Обработывает предупреждения с помощью внутреннего плагина
- warning_recorded**
Обработывает предупреждения с помощью внутреннего плагина
- runtest_logreport**
Реализует `_pytest.reports.TestReport` produced для каждой из трёх фаз (setup, call, teardown) тест-кейса
- assertrepr_compare**
Возвращает объяснения для сравнений в непрошедших assert
- assertion_pass**
(экспериментально) Вызывает ся, когда assert проходит

Debugging

- internalerror**
В случае внутренней ошибки
- keyboard_interrupt**
В случае клавиатурного прерывания
- exception_interact**
Вызывается при обработке исключения. Для внутренних исключений (нпр., `skip.Exception`) не работает
- enter_pdb**
Перед входом в отладочный режим pdb

collection

- начиная с Session как с иницилирующего коллектора:
- `pytest.collectstart(collector)`
 - `report = pytest.make_collect_report(collector)`
 - `pytest.exception_interact(collector, call, report)`
если поднялось интерактивное исключение
- для каждого найденного узла:
- если тест-кейс (item), то `pytest.itemcollected(item)`
а если коллектор, то рекурсивный вызов
 - `pytest.collectreport(report)`
 - `pytest.collection_modifyitems(session, config, items)`
 - `pytest.deselected(items)`
для каждого невыбранного тест-кейса (может быть вызвано много раз)
 - `pytest.collection_finish(session)`
- `session.items = [найденные тест-кейсы (item)]`
- `session.testscollected = количество найденных тест-кейсов`

runtest_protocol

- Setup**
- `pytest.runtest_logstart(nodeid, location)`
 - `call = pytest.runtest_setup(item)`
обёрнуто в `CallInfo(when="setup")`
 - `report = pytest.runtest_makereport(item, call)`
 - `pytest.runtest_logreport(report)`
 - `pytest.exception_interact(call, report)`
если поднялось интерактивное исключение
- если прошла фаза setup и pytest-опция `setuponly` не установлена
- `call = pytest.runtest_setup(item)`
обёрнуто в `CallInfo(when="call")`
 - `report = pytest.runtest_makereport(item, call)`
 - `pytest.runtest_logreport(report)`
 - `pytest.exception_interact(call, report)`
если поднялось интерактивное исключение
- если прошла фаза setup и pytest-опция `setuponly` не установлена
- `call = pytest.runtest_setup(item)`
обёрнуто в `CallInfo(when="teardown")`
 - `report = pytest.runtest_makereport(item, call)`
 - `pytest.runtest_logreport(report)`
 - `pytest.exception_interact(call, report)`
если поднялось интерактивное исключение
 - `pytest.runtest_logfinish(nodeid, location)`

