## Команды

## pip freeze > requirements.txt

## Эта команда сохранит все версии пакетов в специальный файл **requirements.txt.**

pip install -r requirements.txt

В свежем окружении все пакеты установлены одной командой!

## pytest scripts/selenium\_scripts # найти все тесты в директории scripts/selenium\_scripts

## pytest test\_user\_interface.py # найти и выполнить все тесты в файле

## pytest scripts/drafts.py::test\_register\_new\_user\_parametrized # найти тест с именем test\_register\_new\_user\_parametrized в указанном файле в указанной директории и выполнить

## - v – verbose – подробный отчёт

## pytest –v test\_abs.py

## Другие полезные команды для манипуляции выводом тестов PyTest можно найти по ссылке: [Useful py.test commands.](https://gist.github.com/amatellanes/12136508b816469678c2) (<https://gist.github.com/amatellanes/12136508b816469678c2>)

указать параметр **-s**, чтобы увидеть текст, который выводится командой print().

pytest -s test\_fixture1.py

## pytest -s -v test\_fixture2.py

## pytest -s -v -m smoke test\_fixture8.py

## pytest -s -v -m "not smoke" test\_fixture8.py

## pytest -s -v -m "smoke or regression" test\_fixture8.py

## pytest -s -v -m "smoke and win10" test\_fixture81.py

## smoke - метка критичных тестов,  которые нужно запускать на каждый коммит разработчиков

## regression – метка регрессионных тестов, которые запускаются только перед релизом

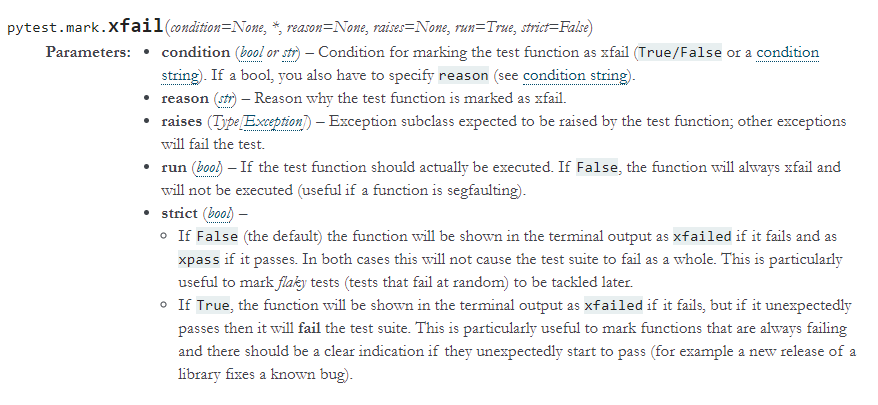
## **@pytest.mark.skip** - метка, которой отмечают в коде, чтобы пропустить тест

## @pytest.mark.xfail

## @pytest.mark.xfail(reason="fixing this bug right now")

## pytest -rx -v test\_fixture10a.py -->> -rx – для xfail+reason

## pytest -rX -v test\_fixture10b.py -->> X в параметре -r, чтобы получить подробную информацию по XPASS-тестам



сondition - (условие) - условие при котором тестовая функция помечается как xfail (не работающая) , принимает либо булево значение либо строку, при булевом значении, нужно будет указать дополнительный параметр reason(причина)

reason - (причина) - причина по которой функция помечена как xfail, принимается строка

raises - (выбрасывания исключений) подкласс исключений которые ожидаются при вызове теста, другие исключения завалят тест

run - (запуск) - параметр отвечающий за запуск теста, при значении False будет всегда xfail и не будет запускаться. принимается булевы значения

strict - (строгость) значение False по умолчанию,  в выводе терминала  будет xfailed, если функция провалила тест и xpass, если прошла. при значении True в выводе терминала будет xfailed, если функция не прошла тест, но если фунцкция неожиданно прошла тест то будет fail

## <https://pytest-docs-ru.readthedocs.io/ru/latest/skipping.html>

## xfail (x): Тест не должен был пройти, был запущен и провалился. XPASS (X): Тест не должен был пройти, был запущен и прошел! Вроде разные понятия, но один параметр xfail с False это запретить

Для себя сделал шпаргалку, что бы все в одном месте по маркировке и пропускам(постарался вместить основное на мой взгляд)  
После маркировок @pytest.mark..., через двоеточие указывал команду для запуска скрипта  
Может кому тоже пригодится..  
  
**Маркировка тестов и пропуск тестов**

**# Запустится только smoke:**  
@pytest.mark.smoke : pytest -s -v -m smoke test\_fixture.py

**# Запуска всех тестов, не отмеченных как smoke:**  
@pytest.mark.smoke : pytest -s -v -m "not smoke" test\_fixture.py

**# Запуск тестов с разными маркировками:**  
@pytest.mark.smoke  
@pytest.mark.regression : pytest -s -v -m "smoke or regression" test\_fixture.py

**# Запуск тестов имеющих несколько маркировок:**  
@pytest.mark.smoke  
@pytest.mark.win10 : pytest -s -v -m "smoke and win10" test\_fixture.py

**# Пропуск тестов:**  
@pytest.mark.skip : pytest -s -v  test\_fixture.py

**# Помечать тест как ожидаемо падающий(пометка:XFAIL):  
# упавший тест теперь отмечен как xfail, но результат прогона тестов помечен как успешный  
# Когда баг починят, мы это узнаем, так как тест будет отмечен как XPASS**  
@pytest.mark.xfail : pytest -rx -v test\_fixture.py

**# reason - Чтобы увидеть это сообщение в консоли, при запуске нужно добавлять параметр pytest -rX**  
@pytest.mark.xfail(reason="fixing this bug right now") : pytest -rX -v test\_fixture.py

# Параметр **strict**  
# Ни XFAIL, ни XPASS по умолчанию не приводят к падению всего набора тестов.  
# Но это можно изменить, установив параметру strict значение True:  
# В этом случае, если тест будет неожиданно пройден (XPASS),  
# то это приведет к падению всего тестового набора  
@pytest.mark.xfail(strict=True) : pytest -rX -v test\_fixture.py

**Область видимости scope**

Для фикстур можно задавать область покрытия фикстур.  
Допустимые значения: “**function**”, “**class**”, “**module**”, “**session**”.

## Запускать тесты, указывая при запуске параметр browser\_name, такой командой:

## pytest -s -v --browser\_name=firefox test\_cmd.py

## **"**--reruns n**"**, где n — это количество перезапусков

## параметр ****"--tb=line"**,** чтобы сократить лог с результатами теста --tb=line - указывает, что нужно выводить только одну строку из лога каждого упавшего теста.

## pytest -v --tb=line --reruns 1 --browser\_name=chrome test\_rerun.py

## <https://docs.pytest.org/en/latest/contents.html>

## <https://www.youtube.com/watch?v=2-EBSIRs0H4&t=1464s>

## Полезные ссылки

### Git

* <https://learngitbranching.js.org/> — отличный интерактивный туториал
* <https://git-scm.com/book/ru/v2/> — лучшая книга вообще
* <https://hyperskill.org/learn/topic/257/>﻿
* <https://stepik.org/course/4138/>﻿
* <http://www-cs-students.stanford.edu/~blynn/gitmagic/intl/ru/index.html>
* <https://habr.com/company/intel/blog/344962/>
* <https://githowto.com/ru>

### Тестирование веб-приложений

* <https://realpython.com/python-testing/> — инструменты для тестирования кода в Python
* [Пирамида тестов на практике](https://habr.com/ru/post/358950/)
* [unittest — документация](https://docs.python.org/3/library/unittest.html)

### Тестирование с помощью PyTest

* [Статья про PyTest](https://habr.com/ru/post/269759/)
* [Введение в PyTest](https://medium.com/@dmrlx/%D0%B2%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B2-pytest-cc6175c7d0dc)
* [Документация PyTest](https://docs.pytest.org/en/latest/)
* [Conventions for Python test discovery](https://docs.pytest.org/en/stable/goodpractices.html)
* [Полезные флаги pytest](https://gist.github.com/amatellanes/12136508b816469678c2)

### Использование фикстур в PyTest

* [Фикстуры — определение](https://en.wikipedia.org/wiki/Test_fixture#Software)
* [Фикстуры в PyTest](https://docs.pytest.org/en/stable/fixture.html)
* [setup и teardown методы](https://docs.pytest.org/en/stable/xunit_setup.html)
* <https://habr.com/ru/company/yandex/blog/242795/>
* <https://medium.com/@dmrlx/%D0%B2%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%B2-pytest-cc6175c7d0dc>
* [Skip and xfail: dealing with tests that cannot succeed](https://pytest.org/en/stable/skipping.html)

### Параметризация, конфигурирование, плагины

* [Parametrizing fixtures and test functions](https://docs.pytest.org/en/stable/parametrize.html)
* [Инструкции по установке geckodriver](https://selenium-python.com/install-geckodriver)
* [Передача параметров в PyTest из командной строки](https://docs.pytest.org/en/stable/example/simple.html?highlight=addoption)
* [Список плагинов PyTest](http://docs.pytest.org/en/stable/plugins.html)
* [Полный список доступных плагинов с описаниями](https://docs.pytest.org/en/latest/reference/plugin_list.html)
* [Настройка вывода PyTest](https://docs.pytest.org/en/stable/usage.html#modifying-python-traceback-printing)