LU03d - Testresultate lesen

Auf einen Blick
PyTest fasst nach jedem Durchlauf zusammen, wie viele Tests bestanden haben. Bei einem fehlgeschlagenen Test zeigt es die Zeile mit dem `assert`, das IST und das SOLL. Wer diese Ausgabe lesen kann, findet Fehler schnell und kann entscheiden, ob der Code oder der Test falsch ist.

Lernziele

Sie können

- die Resultate `passed`, `failed` und `error` unterscheiden,
- in der Ausgabe eines fehlgeschlagenen Tests IST und SOLL finden,
- entscheiden, ob ein Fehler im Programm oder im Test liegt.

Die Zusammenfassung

Wir führen die Tests aus LU03b aus. Ein dritter Test wurde ergänzt und enthält einen Fehler:

```
def test_one():
    result = factorial(1)
    assert result == 2
```

```
$ pytest
============================= test session starts
==============================
collected 3 items

test_factorial.py ..F
[100%]

================================== FAILURES
==================================
_____________________________________ test_one
_____________________________________

    def test_one():
        result = factorial(1)
>       assert result == 2
E       assert 1 == 2

test_factorial.py:16: AssertionError
========================== short test summary info
==========================
```

```
FAILED test_factorial.py::test_one - assert 1 == 2
========================= 1 failed, 2 passed in 0.03s
==========================
```

Teil der Ausgabe	Bedeutung
`collected 3 items`	PyTest hat 3 Testfunktionen gefunden. Stimmt die Zahl nicht, ist ein Test falsch benannt.
`test_factorial.py ..F`	Ein Zeichen pro Test, in der Reihenfolge der Datei.
FAILURES	Für jeden fehlgeschlagenen Test ein eigener Abschnitt.
>	Diese Zeile hat den Fehler ausgelöst.
`E assert 1 == 2`	Links steht das **IST** (`factorial(1)` ergab 1), rechts das **SOLL** aus dem Test (2).
`test_factorial.py:16`	Datei und Zeilennummer des fehlgeschlagenen `assert`.
`1 failed, 2 passed`	Das Gesamtergebnis des Durchlaufs.

Die möglichen Resultate

Zeichen	Resultat	Bedeutung
.	passed	Alle `assert` sind erfüllt.
F	failed	Ein `assert` ist nicht erfüllt, oder im getesteten Code ist eine unerwartete Exception aufgetreten.
E	error	Der Test konnte gar nicht richtig laufen, zum Beispiel weil der Import oder eine Fixture fehlschlägt.
s	skipped	Der Test wurde absichtlich übersprungen (LU04c).
x	xfailed	Der Test ist wie erwartet fehlgeschlagen (LU04d).
X	xpassed	Ein Test, der fehlschlagen sollte, hat bestanden.

Mit `pytest -v` erscheint statt der Zeichen jeder Test auf einer eigenen Zeile:

```
test_factorial.py::test_normal PASSED                                    [
33%]
test_factorial.py::test_zero PASSED                                      [
66%]
test_factorial.py::test_one FAILED
[100%]
```

Fehler beim Sammeln der Tests

Kann PyTest eine Testdatei nicht einmal importieren, läuft kein einziger Test daraus:

```
test_import.py:1: in <module>
    from factorial import fakultaet
E   ImportError: cannot import name 'fakultaet' from 'factorial'
=========================== short test summary info
============================
ERROR test_import.py
!!!!!!!!!!!!!!!!!!!! Interrupted: 1 error during collection
```

```
!!!!!!!!!!!!!!!!!!!
```

Hier liegt der Fehler nicht in der Logik, sondern im Testcode: Die Funktion heisst `factorial`, nicht `fakultaet`.

Listen und Texte vergleichen

Bei Listen, Tupeln und Texten zeigt PyTest genau, wo sich IST und SOLL unterscheiden:

```
>       assert [1, 2, 3] == [1, 2, 4]
E       assert [1, 2, 3] == [1, 2, 4]
E
E         At index 2 diff: 3 != 4
```

Wo liegt der Fehler?

Ein roter Test heisst nur: IST und SOLL stimmen nicht überein. Bevor Sie den Code ändern, prüfen Sie beide Seiten:

1. **Stimmt das SOLL?** Rechnen Sie den erwarteten Wert von Hand nach oder lesen Sie die Anforderung nochmals. Im Beispiel oben ist `1! = 1`. Der **Test** ist falsch, nicht die Funktion.
2. **Stimmt das IST?** Ist das SOLL korrekt, liegt der Fehler im Programm. Die Zeilennummer und das IST helfen, die Stelle zu finden.
3. **Mehrere Tests rot?** Schlagen alle Tests mit einer leeren Liste oder mit 0 fehl, ist es wahrscheinlich **ein** Fehler in der Behandlung dieses Spezialfalls. Suchen Sie nach dem Muster, bevor Sie einzelne Tests anschauen.

Passen Sie nie das SOLL an das IST an, nur damit der Test grün wird. Ein Test, der immer besteht, findet keine Fehler.

Kontrollfragen

1. PyTest meldet `collected 4 items`, Ihre Datei enthält aber fünf Testfunktionen. Woran kann das liegen?
2. Was ist der Unterschied zwischen F und E in der Ausgabe?
3. In der Ausgabe steht `E assert 120 == 24`. Welcher Wert ist das IST, welcher das SOLL?
4. Nach einer Änderung schlagen alle Tests mit negativen Zahlen fehl, alle anderen bestehen. Was schliessen Sie daraus?

M450-LU03, M450-D2B

Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m450/learningunits/lu03/resultate

Last update: **2026/11/17 07:57**