

Teszt vezérelt fejlesztés

Hozzunk létre egy könyvtárat és lépünk bele:

```
mkdir tdd  
cd tdd
```

Hozzunk létre egy virtuális környezetet:

```
python -m virtualenv .venv
```

Ha nincs telepítve a *virtualenv* csomag akkor futtassuk:

```
pip install virtualenv
```

Lépünk be a virtualis környezetbe:

```
.\.venv\Scripts\activate
```

A prompt előtt inentől kezdve (.venv) szöveg íródik ki. Telepítsük a *pytest* csomagot:

```
pip install pytest
```

A teszt indítása:

```
python -m pytest tests
```

Minden könyvtárban keresi a *test_*.py* és a **_test.py* fájlokat, ezeket tekinti teszteknek.

Teszt írása

hozzunk létre egy *test_basic.py* fájlt a teszt könyvtárban a következő tartalommal:

```
def test_always_passes():  
    assert True  
  
def test_always_fails():  
    assert False
```

Majd futtassuk a tesztet:

```
python -m pytest tests
```

Hozzunk létre egy másik teszt fájlt, *test_fixture.py*:

```
import pytest

@pytest.fixture
def example_fixture():
    return 1

def test_with_fixture(example_fixture):
    assert example_fixture == 1
```

A teszt futtatása után az alábbi eredményt kapjuk:

```
(.venv) c:\projects\tdd>python -m pytest tests
===== test session starts =====
platform win32 -- Python 3.10.11, pytest-8.0.2, pluggy-1.4.0
rootdir: c:\projects\tdd
collected 3 items

tests\test_basic.py .F [ 66%]
tests\test_fixture.py . [100%]

===== FAILURES =====
_____ test_always_fails _____

    def test_always_fails():
>         assert False
E         assert False

tests\test_basic.py:7: AssertionError
===== short test summary info =====
FAILED tests/test_basic.py::test_always_fails - assert False
===== 1 failed, 2 passed in 0.04s =====
```

Javítsuk ki a tesztet és github repositoryba töltsük fel.

Hozzunk létre egy app könyvtárat a gyökérben és üres `__init__.py` fájlt és egy `data_formatter.py` fájlt, a `tests/test_data_formatter.py`.

```
projekt_könyvtár/
├── app/
│   ├── __init__.py
│   └── data_formatter.py
└── tests/
    └── test_data_formatter.py
```

A `test_data_formatter.py` tartalma legyen:

```
from app.data_formatter import format_data_for_display
def test_format_data_for_display():
    people = [
        {
```

```
        "given_name": "Karoly",
        "family_name": "Nehez",
        "title": "Senior Software Engineer",
    },
    {
        "given_name": "John",
        "family_name": "Smith",
        "title": "Project Manager",
    },
]

assert format_data_for_display(people) == [
    "Karoly Nehez: Senior Software Engineer",
    "John Smith: Project Manager",
]
```

A data_formatter.py-ben megírjuk az implementációt:

```
def format_data_for_display(people):
    return [f"{person['given_name']} {person['family_name']}: {person['title']}" for person in people]
```

Futtassuk a tesztet:

```
python -m pytest tests
```

From:

<https://edu.iit.uni-miskolc.hu/> - Institute of Information Science - University of Miskolc

Permanent link:

https://edu.iit.uni-miskolc.hu/tanszek:oktatas:informacios_rendszerek_integralasa:tdd_pelda

Last update: **2024/02/25 22:32**

