

# Egységtesztek...

---

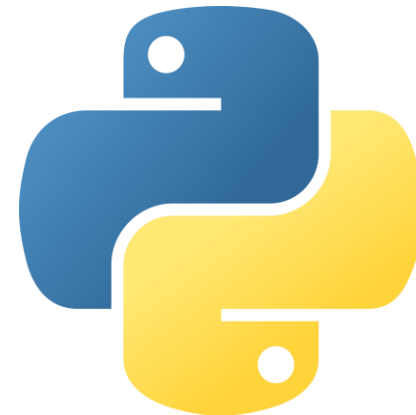
BEBESI LÁSZLÓ

2025.12.04

# Mire jók az egységtesztek?

- Kódminőség javulás
- Korai hiba felismerés
- Dokumentáció egy forrása
- Hatékonyság növelés
- Kód struktúra rögzítése
- Refaktorálás biztonsága / kód karbantartás könnyítése

*Egyebek: Bizalom növekedése? Jobb kódolási gyakorlat? Közös megértés a fejlesztők között?*



# Elvárások az egységtesztekkel szemben

## Gyors

Ideális esetben másodpercek alatt lefut.

## Egységnyi funkció

Ideális esetben egy teszt csak egy kisebb funkciót fed le.

## Önálló

Nem kapcsolódik külső adatforrásokhoz.

## Automatizálható

CI/CD -> Jenkins, Azure devops, GitLab?

## Olvashatóság

Könnyen, egyszerűen megérthető...

## Lefedettség

Az egységtesztek minden lehetséges esetet lefednek

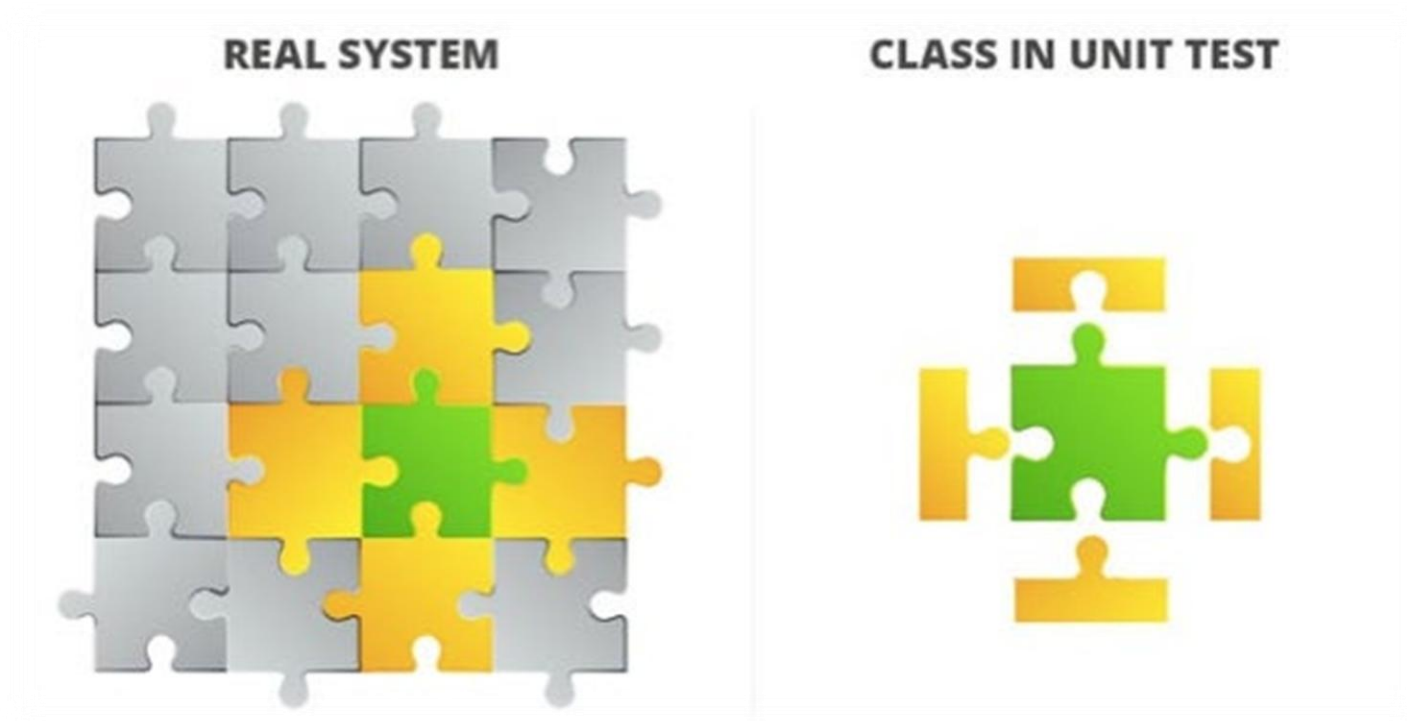
## Szétválasztott

Adott egységteszt nem függ másik unit tesztől...

## Determinisztikus

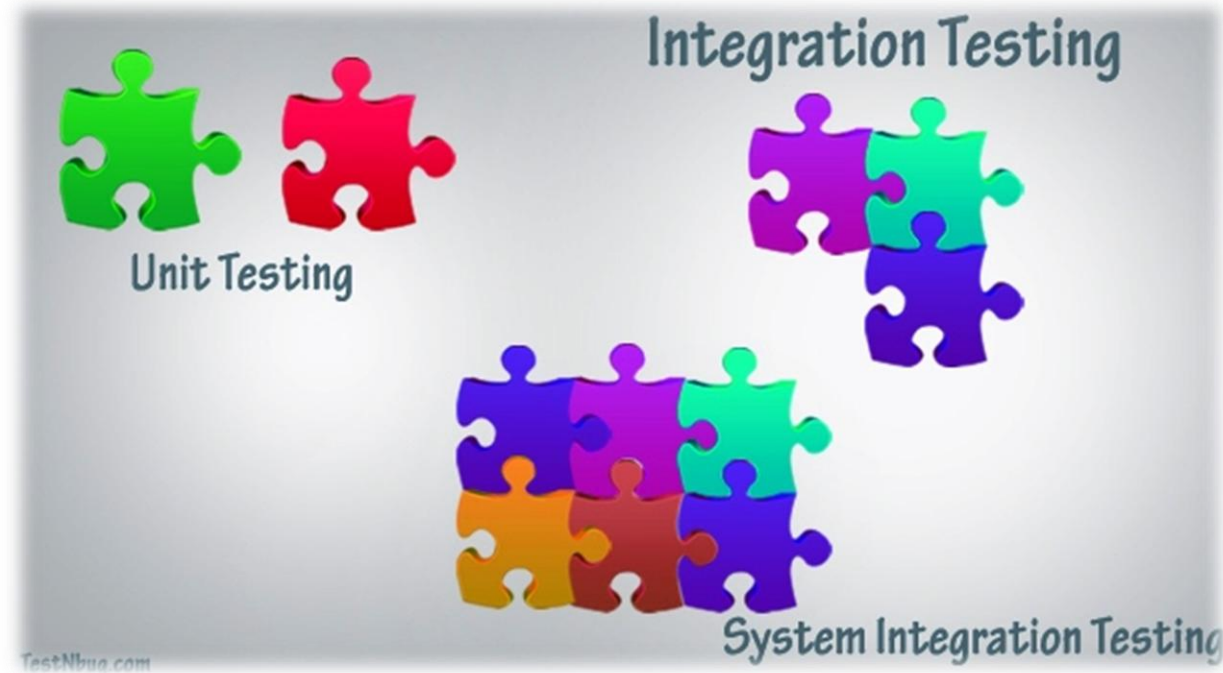
A teszt megismételhető, minden futásnál ugyanazt az eredményt adja.

## *Egységtesztek szemléltetés*



Mivel lehet elérni, hogy leválasszuk a rendszer többi elemét a tesztelendő funkcióról?

## *Integrációs teszt szemléltetés*



A cél az egyes komponensek, függvények stb. együttműködésének tesztelése.  
A fókuszban az egymáshoz közel álló modulok együttműködése áll.

- ➔ pytest, unittest és Django test csomagok használhatóak ezekre is.
- ➔ Djangóban gyakori: „request → view → model → DB → response” teljes lánc tesztelés

*A feladat:*

[https://github.com/bebesi33/django\\_2025\\_fall/blob/main/lecture\\_10/loan\\_calc/scripts/explore\\_loan\\_calc.ipynb](https://github.com/bebesi33/django_2025_fall/blob/main/lecture_10/loan_calc/scripts/explore_loan_calc.ipynb)

(A szokásos GitHub repository 10. gyakorhoz tartozó mappájában ismerkedjünk a loan\_calc package-el.)