**Mi a szoftvertesztelés?**

Ha egy mondatban kellene megválaszolnunk a fenti kérdést, ilyen választ adhatnánk rá: **a szoftvertesztelés a szoftverfejlesztési folyamatot tervezetten kísérő, azzal szorosan összefüggő, ám annak önálló része, melynek célja a tesztelt szoftver minőségének feltárása**.

A szoftverek tesztelését megtervező, végrehajtó és az eredményeket kiértékelő szakembert **szoftvertesztelőnek**, **tesztmérnöknek** nevezik (ez angolul tester, ill. test engineer).

## . A tesztelés célja

A szoftvertesztelés egyik célja a

* **hibák bekövetkeztének csökkentése, vagy kiküszöbölése**.

Másik célja

* olyan információk előállítása, melyek alapján kijelentéseket lehet **megfogalmazni a szoftver minőségéről és döntéseket lehet hozni a fejlesztéssel kapcsolatban**

# 2. A tesztelés alapfogalmai

A szoftvertesztelés alapfogalmai alapvetően angol fogalmak, így magyarításuk nehezen, vagy nem lehetséges.

Ennek figyelembevételével ismertetjük a szoftvertesztelés alapvető fogalmait.

## A hiba fogalma

Miért tartalmazhatnak a szoftverek hibákat? A válasz egyszerű: mert emberek készítik őket. Az ember pedig esendő és semmiképpen sem tévedhetetlen. (Valójában a számítógépek sem azok, de szeretjük azt hinni, hogy tévedhetetlenek.)

Ismerjük meg a hibák különböző fogalmait!

**Hibázás** (error, mistake): ez a hibafogalom egy hibajelenség létrehozásának, létrejöttének a folyamatát takarja.

**Hiba** (defect, fault, bug): konkrét, kimutatható hibajelenség. **A tesztelés elsődleges célja a hibák megtalálása és dokumentálása.**

**Hibás működés** (failure): a hibajelenségből eredő, az elvárttól eltérő működés, melyet nem emberi tényezők is okozhatnak (például, amikor áramszünet miatt leállnak szerverek).

Az ún. **debugging** során a programozó felügyelt módban képes futtatni a szoftverét és képes a szoftver alkotóelemeinek ellenőrzésére. Tesztelésnek számít-e ez a fajta hibakeresés? A válasz egyértelműen: **nem**. A debugging a fejlesztési munkafolyamat fontos része, mely a forráskód megírását és a futtatható szoftver létrehozását, valamint javítását segíti. A tesztelő azonban nem így dolgozik.

## Teszt

A **tesztelés** az a tevékenység, melynek során - előre meghatározott feltételek teljesülése esetén - a teszt tárgyát képező szoftver adott tulajdonságát szintén előre meghatározott ismérvek és feltételrendszerek alapján megvizsgáljuk, elemezzük, dokumentáljuk és összefoglaljuk. A teszttevékenység végrehajtását **tesztvégrehajtásnak**, **tesztfuttatásnak (test run)** nevezzük.

## Tesztkörnyezet

A tesztkörnyezet egy olyan informatikai infrastruktúra, mely a leendő, vagy már meglévő éles rendszerinfrastruktúrát modellezi, ill. a felhasználás körülményeit képes szimulálni.

Tesztkörnyezet állhat egyetlen számítógépből, de akár több kontinensen szétszórt, hatalmas számítógépközpontokból is.

Ideális esetben a tesztkörnyezet tulajdonságai megegyeznek az éles környezetével. A költségek kímélése végett a valóságban az éles rendszer paramétereihez közeli tesztkörnyezeteket szoktak létrehozni.

Ez általában elegendő is szokott lenni.

**Éles rendszert tesztelésre használni tilos!**

### Üzleti követelmény (business requirement)

Az üzleti követelményeket nem a tesztelők állítják fel, hanem az üzleti részleg felelősei. Egy tesztprojektben kell lennie egy **üzleti elemzőnek**, akinek az is feladata, hogy az üzleti oldal igényeit ún. **üzleti folyamatokra** bontsa.

Ilyen üzleti folyamat lehet egy banki szoftver esetében például egy ügyfél hitelkérelmének elfogadása.

Az üzleti követelmények soha nem technikai leírások, hanem folyamatleírások.

## Felhasználási folyamatleírások (user story)

Ezen leírások részletessége változó, de gyakran hasonlóak a tesztesetek tesztlépéseinek leírásához (lásd lentebb).

Megfogalmaznak feltételeket, tevékenységeket és ún. **elvárt eredményeket**.

## Teszt szcenárió (test scenario)

A teszt szcenáriók az üzleti követelményekből levezetett összefoglaló leírások.

Itt jegyezzük meg, hogy a teszt szcenárió, felhasználási eset és üzleti követelmény a gyakorlatban nem feltétlenül különálló leírásokat jelent. Erre egységesen alkalmazott rendszer a gyakorlatban nincsen, projektenként változhat a megvalósításuk.

### Teszteset (test case)

A tesztesetek a tesztvégrehajtás során elvégzendő, logikailag összetartozó teszttevékenységek összességét jelentik.

A teszteseteket különböző névkonvenciók alapján nevekkel szokás ellátni, melyek utalnak a teszt tárgyára.

Például: Test\_#1\_Új\_ügyfél\_felvétele

### Tesztlépés (test step)

A **tesztesetek egyes elemi tevékenységeit** tesztlépésnek nevezzük.

Nagyon fontos, hogy a tesztlépés valóban elemi tevékenységek leírása legyen, azaz további lépésekre ne legyen bontható.

Ennek nagyon egyszerű oka van. Ha az adott tesztlépés több elemi lépést is tartalmazna és hibát találunk, a hiba visszakeresését és visszakövethetőségét nagymértékben rontaná, ha nem egyetlen elemi lépést, hanem lépések halmazát kellene ismét megvizsgálnia a tesztelőnek, vagy a hiba javítását végző fejlesztőnek.

Az elemi tesztlépések belső szerkezetüket tekintve további egységekre bonthatóak, melyek maguk is leírások. A két legfontosabb egység:

Magának a végrehajtandó tevékenységnek a leírása.

Az ún. elvárt eredmény megfogalmazása. Az **elvárt eredmény** annak a leírása, hogy mi történik akkor, ha a tesztlépés végrehajtása **sikeres** volt, tehát nem lépett fel hiba.

A tesztlépések tartalmazhatnak további információkat is:

Tesztadatok megadása.

További megjegyzések a teszt végrehajtásához.

Fontos tudni, hogy egyes projektekben a folyamatok hangsúlyosabbak, másokban a tesztadatok, ill. a konkrét teszteredmények állnak középpontban.

## Review

A review angol szakkifejezés és felülvizsgálatot jelent. A programozás során is használt módszer segítségével növelhető a tesztesetek minősége.

A review során a tesztelő az általa elkészített tesztesetet átadja egy társának, aki leellenőrzi, hogy a teszteset valóban megfelel-e minden követelménynek és minden szükséges információt tartalmaz-e.

A módszer hatékonyan alkalmazható hasonló és eltérő tapasztalattal rendelkező tesztelőknél is. A gyakorlati kivitelezése lehet informális, de akár erősen formalizált, írásban rögíztett is.

A felülvizsgálat végén a teszteset eredeti készítője megkapja mindazon hiányosságok listáját, melyeket még be kell építenie a tesztesetbe.

Egy teszteset elviekben csak review után használható éles tesztelésre.

## Teszt státusz

A tesztek státusza egy fogalom, elnevezés, mely röviden összefoglalja és megadja az adott tesztesethez tartozó teszt aktuális állapotát.

Néhány példa tesztek státuszaira:

Előkészületben (In preparation)

Végrehajtható (Ready)

Folyamatban (In Progress)

Blokkolt (Blocked)

Hibás tesztfuttatás (Failed)

Sikeres tesztfuttatás (Passed)

Nem végrehajtható (Not executable)

Nem alkalmazható (Not applicable)

A státuszok száma és elnevezése erősen projektfüggő. A státuszok sorrendjét szokás és ajánlatos előre meghatározni is, ez segíthet nagyszámú **hibajelentés** feldolgozásában. Folyamatábrával áttekinthető, hogy egyik státusz után milyen másik következik.

## Negatív tesztesetek

Átfogó tesztelés során nem csupán valaminek a sikerességét kell ellenőrizni, hanem a szoftver hibatűrését is.

Ilyenkor a tesztelő kifejezetten „kiprovokálja” a szoftver reakcióját a hibára.

Ezeket a teszteseteket nevezzük **negatív célzatú teszteseteknek**. Tipikusan ilyen tesztek a biztonsági tesztek, melyeket kifejezetten erre szakosodott tesztelők végeznek.

Úgy is mondhatnánk, hogy a negatív teszteseteknél az a hiba, ha nincs hiba.

## A tesztelési folyamat főbb állomásai

Ideális esetben a tesztelés legalább a következő, alapvető fázisokból áll:

Tervezés

Elemzés

Tesztdizájn

Végrehajtás

Kiértékelés

Tesztjelentés elkészítése

A fázisok ugyan átfedésbe kerülhetnek és ismétlődhetnek is, ám egyfajta logikai hierarchia mégis fennáll közöttük. Ily módon egy fázis gondatlan, vagy hanyag előkészítése negatívan befolyásolhatja az összes többi fázis hatékonyságát is.

### A tesztek megtervezése

A tesztek végrehajtását gondos tervezés kell, hogy megelőzze. A tervezés során meghatározzák a teszt célját, a teszt tárgyát, a teszttevékenységek végrehajtásának feltételeit, módját és a kiértékelés kritériumait.

A tervezés kimenete általában egy vagy több dokumentum, melyet a tesztmenedzser, vagy a vezető tesztelő készít el, vagy éppen mindketten.

A tesztterv nem részletes technikai megvalósításokat taglal, hanem folyamatokat ír le, témaköröket mutat be.

A tervezés nem csak a tesztelők, hanem a fejlesztők és a szoftver megrendelője számára is készül. A fejlesztők megtudhatják belőle, hogyan és mire alapozva fognak hibajelentéseket kapni a tesztelőktől, a megrendelő pedig megbizonyosodhat afelől, hogy a szoftvert minden, általa fontosnak ítélt szempont szerint le fogják tesztelni.

A tesztterv nem egy „kőbe vésett” dokumentum, hanem alakítható a résztvevők javaslatai alapján.

Súlyos szakmai hiba, ha a tesztterv elkészítője nem veszi figyelembe például a fejlesztőknek a tesztelés ütemezésére vonatkozó észrevételeit, módosító javaslatait.

### Teszt tervezés sprintekben

Tesztek tervezésére nem csak projektek induló fázisában, hanem sprintekben (SCRUM rendszerben, lásd később) is szükség van. Ez egy kevésbé formalizált tervezés, inkább egy kötetlen szakmai beszélgetés például az üzleti elemző, a projekt menedzser, a fejlesztők és természetesen a tesztelők között.

A tesztelők felelőssége, hogy ilyenkor minden kérdést tisztázzanak, amely a tesztelés végrehajtását veszélyeztetheti. Ennek során általában olyan kérdések is felmerülnek, melyek a fejlesztők számára is létfontosságúak, tehát az eljárással minden résztvevő jól jár.

Ha ez a megbeszélés elmarad, az sprint közben többnyire megakasztja a tesztelés és nem ritkán a fejlesztés lendületét is. A tesztelők információk híján nem tudják pontosan ellátni feladatukat, menet közben kell tisztázni részleteket. Emiatt rengeteg idő elmehet, ami végeredményben többletköltséget eredményez.

### A tesztelemzés

A teszt végrehajtóinak a tesztterv magas szinten megfogalmazott céljait és végrehajtásának feltételeit tételesen meg kell vizsgálniuk és számba kell venniük, hogy a rendelkezésre álló információk alapján a teszt kivitelezhető-e.

Az elemzés során a teszt egyes részeinek a priorizálása is megtörténik.

Sajnos a gyakorlatban elhanyagolt mozzanat szokott lenni a tesztkörnyezet előkészítése, illetve az esetlegesen beszerzendő teszteszközök feltérképezése is. A folyamat végén ad absurdum a tesztkörnyezet létrehozása is megkérdőjeleződhet, ami az egész fejlesztési projektet ahhoz teszi hasonlóvá, amit „orosz rulett” néven ismernek az emberek.

Az elemzési fázis nem öncélú dolog, hiszen képes részletekbe menően felfedni az adott szoftver tesztelhetőségét.

Ez szélsőséges esetben azt is jelentheti, hogy az elemzés azt tárja fel, hogy a célként megfogalmazott tesztelés az előzetesen megállapított feltételekkel nem hajtható végre.

### Tesztdizájn, előkészítés

A tesztdizájn során a végrehajtandó tesztek a végrehajtandó tesztlépések szintjén lesznek megfogalmazva.

Ezek már nem magasszintű megfogalmazások, hanem konkrét tesztlépések szintjén leírt tevékenységek. Lényegében ekkor jönnek létre „fizikailag” is a tesztesetek.

Fontos az esetlegesen szükséges tesztadatok meghatározása és előállítása, automatizált tesztek esetében a teszt szkriptek megírása, ill. az automatizálást segítő szoftver beállítása.

### Tesztvégrehajtás

A tesztek végrehajtása (lefuttatása) természetesen mindig „az igazság pillanata”.

Minden tesztvégrehajtásnak van eredménye. Még az is eredmény, ha a teszt maga nem hajtható végre.

A végrehajtás végén mindig a kiértékelés és az eredmények dokumentálása is megtörténik.

### Tesztkiértékelés

A kiértékelés során a dizájn fázisban megfogalmazott feltételek és a szintén akkor megfogalmazott, ún. elvárt teszteredmények összehasonlítása történik meg, a tényleges tesztvégrehajtás eredményeivel.

A kiértékelés egy rövid összegző jelentés formájában is testet ölt, melyet **teszt riportnak** is neveznek.

A tesztriport egyik változata a **záró teszt riport** (test exit report) mely a tesztelés egészét értékeli ki, annak teljes végrehajtása után és prezentálja a teszteredményeket is.

### Tesztelést támogató tevékenységek

A tesztelés menedzselése során gyakran elfeledkeznek a tesztkörnyezetek karbantartásáról és a tesztadatok előállításáról, pedig mindkét tevékenység alapjaiban befolyásolhatja a tesztelés sikerességét.

A tesztkörnyezet karbantartása kisebb cégek esetében egy-egy számítógép felügyeletét és folyamatos naprakészen tartását jelenti. Különösen fontos, hogy dedikált tesztelői számítógépeket használjanak. Ez azt jelenti, hogy ezeket a gépeket kizárólag tesztelésre szabad használni. Nagyvállalati környezetben a tesztkörnyezetek menedzselése komoyl kihívást jelent, amit jól jellemez az is, hogy egész szakember gárdákat alkalmaznak kizárólag a gépek üzemeltetésére és a szoftvertelepítések koordinálására.

A tesztadatok előállítása első hallásra banálisnak tűnhet. Miért kellene folyamatosan a teszteléshez használt adatokat karbantartani?

A szoftverek fejlődése sajnos ezt kikényszeríti. Ahol erre nem figyelnek oda, ott a tesztelőknek sanyarú sorsa van, ugyanis utólagosan szembesülnek azzal, hogy korábban használt tesztadataik hibás teszteredményeket produkálnak. Az utólagos időráfordítás, nyomozgatás természetesen rengeteg időt vesz el és plusz költségeket okoz.