Az Excel-macro tanítás/tanulás tapasztalatai

(Experiences of teaching/learning of Excel-macros)

Pitlik László, Pitlik Marcell – MY-X team

Kivonat: A programozni tanulás, hasonlóan a nyelvtanuláshoz tévhitek és szerencsés véletlenek által befolyásolt folyamat. Tévhitek azok a megzavaró álszabályszerűségek, melyek a nyelvtanulás esetében egy kevés élőnyelvi tapasztalattal rendelkező oktató által kerülnek kimondásra, de később ezekről bebizonyosodik, hogy nem voltak igazak: pl. az „ezt így nem mondják” elv indokolatlan használata. Szerencsés véletlen a nyelvtanulásban, ha valaki olyan nagyobb tudású személlyel találkozik (megfelelő forrásdokumentumra akad), aki/ami már átélte/leírta a kiképzendőt éppen sújtó anomáliát és képes ezt feloldani azonnal. A programozás esetén a folyamatba belépés gátjaiként értelmezhetők pl. a túl komplex előfeltételek kikényszerítése (pl. macro-zás esetén a DIM használata). A programozás-tanulás során a szerencsés véletlen is azon kulcsinformációkat közölni képes személy/forrás fellelése (pl. hogyan kell eddig dinamikusan paraméterezhetetlennek hitt paraméter-helyet ciklusváltozóval vezérelni). Ebben a cikkben az éppen MS Excel-ben macro-zást tanulni szándékozók, ill. az macro-zást oktatók számára kerülnek bemutatásra és értelmezésre valós tanulási tapasztalatok.

Kulcsszavak: önellenőrzés, előfeltételek minimuma, minőségbiztosítás

Abstract: The effectivity/efficiency of learning processes concerning foreign languages are influenced through wrong rules and/or lucky constellations. Wrong rules or fake news can be integrated into the learning processes if a teacher without appropriate experiences declares that a specific solution/sentence will never be said in the real life – although it is not true. The lucky factor is if the catastrophic impacts of a wrong rule can be eliminated quasi at once based on contacts with persons/sources having the same problem and the adequate solution. In case of creating source codes, wrong rules should be re-defined as too complex starting rules (e.g. using DIM before coding). The lucky effects are the same: if the motivated learners have the opportunity early enough to meet persons/sources where it will be possible to see how the seemingly untouchable parameters/arguments can be controlled in a dynamic way within a for-to-next statement. This article presents cases where anomalies/specialities of real/anonymized learning situations will be described and analysed.

Keywords: self-control, minimum of requirements quality management

# Bevezetés

A MS Excel által felkínált/támogatott macro-rögzítési/írási lehetőségről szubjektív vélemények hosszú időt átfogó sorozata alapján vélelmezhető, hogy egy vonzó belépési pont úm. programozási szempontból még laikus személyek számára a programozás világába. Ennek oka viszonylag egyszerű: a macro rögzítés pszichológiailag megnyugtató kiindulási pontként kerül beazonosításra a laikusok által. Megnyugtató, mert már az első programkód keletkeztetéséhez sem kell tudni alapvetően semmit – magának a macro-rögzítőnek az elindításán/leállításán túl. S ez pedig nem komplexebb tudás, mint bármilyen más Excel-menüpont/függvény értelmezése, használata.

A fekete leves azonban azonnal jön, amint az első saját maguk által rögzített macro-ból célirányosan tovább kellene lépni.

Ebben a cikkben a fenti kiindulási helyzetből kitörést jelentő minimális információmennyiség kerül bemutatásra. Emellett a laikusok hasznos és quasi önsorsrontó magatartásmintái is leírásra kerülnek annak támogatására, hogy az önsorsrontást minél többen minél hamarabb legyenek szívesek elkerülni.

Ez a cikk ellenpontja minden klasszikus programozás-oktatásnak, ahol rel. sok/komoly alapozás után illik csak megalkotni az első kódot.

# Előzmények

Az alábbi állomány egy olyan kompakt macro-írási tudást tartalmaz, melyre elvileg akár autodidakta módon is szinte bárkinek képesnek kell lennie tudni alapozni: <https://miau.my-x.hu/digeco/coco_prg1.xlsm>

A fenti állomány egy rel. komplex számítási logikát mutat be Excel-ben cellaláncokat szintjén és az ezeket előállító macro-k szintjén. Az XLSM állomány első megismerése nem autodidakta tanulási feltételek mellett, hanem valódi, ill. skype-alapú személyes (konktakórát jelentő) találkozás mellett történik meg.

A macro-világ első érzékelése természetesen macro-rögzítés eredményeként előálló kódrészlet közös olvasásával kezdődik (pl. sorozatos cella-formázási lépések), ahol a cél annak demonstrálása, hogy rel. kevés (nulla) tudással a kód szerkezetéről, logikájáról, vagyis a kódot butuska titkosírásnak tekintve, mennyi racionális feltételezéssel lehet élni a valódi érdeklődőnek, aki kiképzendőként nem csak kreditet/papírt akar, hanem át akarja élni a felfedezések valódi örömeit, a szuverenitás szépségét.

A macro-rögzítés előtt azonban már érdemes utalni arra is, hogy minden olyan apróbb/nagyobb támogatás is programnak tekinthető, ami képes repetitív/hosszadalmas lépéssorok kiváltására/felgyorsítására. Vagyis egy keylogger maga is már programot hoz létre akkor, ha olyan billentyű- és/vagy egérmozgások kerülnek pontosan naplózásra, melyek eredménye az, hogy ezeket a billentyű- és/vagy egér-használati akciókat bármikor újra és újra elő lehet hívni, s ezek hatása mindenkor pontosan az lesz, mint ha az ember maga írná le/kattintgatná végig újra és újra ugyanazt a sorrendiséget/tartalmat.

Ez a felvezető gondolat illene, hogy az Excel által rögzített macro-k kapcsán azonnal bekapcsolja a vészjelzést a fejekben, miszerint egy egyszer már valaki által végrehajtott lépéssor mindenkor (minden körülmények között) pontosan a kívánt hatást váltja-e ki? Más szavakkal: mikor tekinthető egy lépéssor olyan intelligensnek, hogy az alkalmazás keretfeltételeinek tételes megadása nélkül is adaptívan reagál arra a környezetre, ahol ezt elindítja az ember.

# Tévhitek, hatékonyságromboló magatartásminták kezdők esetén

Az alábbiakban valós, de anonim történetek alapján ezek esszenciális üzenete kerül megfogalmazásra annak érdekében, hogy lehetőség szerint minden kezdő, aki ezt a dokumentumot olvassa, más kárán tudjon minél nagyobb mértékben és minél hamarabb tanulni. Hasonlóképpen hasznos illene, hogy legyen a felhasználó-közelség ezen esettanulmányszerű nézete az oktatóknak is. Különösen azoknak, akiknek már a hosszú éves tapasztalatok nyomán egyre halványabbak a saját kezdeti tapasztalataik, ill. egyre gyakrabban érzik (ha őszintén bevallják maguknak), hogy egyre kevésbé értik, mit nem lehet érteni mindazon, amit ők fontosnak tartva megosztanak a kiképzendőkkel.

1. Az első eset az autodidakta tanulás egy fajta tipikus kockázatára mutat rá. Az újra és újra ismétlődő történet szereplői olyan nagyon motivált és nagy szuverenitás-vággyal rendelkező személyek, akik úgy gondolják, hogy véletlenszerű oktatóvideók alapján lényegében önállóan be fogják tudni járni a szükséges utakat. Ennek az esetnek a fő kockázati forrása annak meg nem értése, nem pontos értése, ill. félreértése, mit is akar ténylegesen a fellelt demo szerzője: vö. <https://www.youtube.com/watch?v=fkWE2qKq3hI>
   1. a demo állomány alapján hasznos technikai részletek azonosíthatók be a nyomógomb létrehozása kapcsán
   2. de a „Helló-Világ” bemutatott megoldás tökéletesen tévútra viszi azt, aki az Excel-t, mint intelligens kockáspapírt akarja egyre hatékonyabban használni, hiszen a MSGBOX olyan kommunikáció rétege a Visual Basic programnyelvnek, mely tökéletesen független az intelligens kockáspapírtól magától, vagyis semmit nem árul el arról, hogy az Excel munkafüzet munkalapjai és ezek cellái miként címezhetők, olvashatók, írhatók…
   3. az intelligens kockáspapírral való kapcsolat fel sem vétele, ill. azonnali megszakadása lényegében az input-output logikát teszi teljesen zavarossá, vagyis a kezdő máris elveszíti annak az alapérzésnek minden kontúrját, hogy az Excel-cellák tartalma lehet inputja egy-egy algoritmusnak és az Excel-cellák maguk lehetnek az eredményt megjelenítő objektumok…
   4. a másik fontos kockázat a mentési folyamat (xlsm) nem említése, vagyis az input, mint emberi kódírás és az output, mint mentett állomány nem tudatosítása ott, ahol az eredménytermék léte a cél…
   5. a videó bizonyos értelemben szerencsére semmilyen formában nem tér ki arra, hogy az MSGBOX-ról még mennyi mindent lehetne még tudni:



1. ábra: Az MSGBOX egy fajta értelmezéstámogató nézete (forrás: videó)

1. Ha marco-rögzítéssel hozza létre a laikus az első helló-világ-effektust adott munkalap C7-es celláját választva pl. nyíl-billentyűkkel/egérkattintással, akkor ezt az eredményt kapja: Range("C7").Select ill. ActiveCell.FormulaR1C1 = "Hello vilag"
   1. ebben az esetben a C7-es paraméterrel való szembesüléskor azonnal rá kell döbbennie, hogy ezt a paramétert nem fogja tudni semmilyen inputtal vezérelni eddigi tudása alapján
   2. s hasonlóképpen kijózanító, s egyben iránymutató, hogy az objektum-orientált logikában a RANGE és az ACTIVECELL, valamint a SELECT és a FORMULA azonos szerepeket töltenek be
   3. végül az R1C1 paraméterek láttán felmerülhet, hogy itt lehet paraméterezhetően is hivatkozni az intelligens kockáspapírra – DE HOGYAN?
2. Ha ezek után a laikus elveszíti alapvető motivációit, akkor ideális esetben még kérdezni mer, vagy a tananyagként felvillantott XLSM-et kezdi el értelmezni:
   1. ha kérdez a kiképzendő a nála tapasztaltabbaktól, akkor a fórumok logikája értelmében jön a gyors válasz (pl. a C8-as cellába írás kapcsán): Sheets("Munka1").Cells(8, 3) = "Helló világ"
   2. ebben a pillanatban egyrészt elmúlnak a paraméterezhetetlenségi aggályok a cella (sor,oszlop) koordinátái kapcsán
   3. de megmaradnak továbbra is a munkalap nevének láttán
   4. viszont a saját maga által írt kód előállítása során a kiképzendő a 2. ábra szerinti segítséget kaphatja meg, ha
      1. maga ír be mindent
      2. ill. a Sheets-et elfelejtve csak a Cells paramétereivel akar foglalkozni
   5. az index, rowindex, columnindex szavak láttán akár még az is felmerülhet a trükkösebb kiképzendőkben spontán, hogy a „Munka1” munkalapnév mögött illene, hogy legyen valamiféle sorszámozott munkalap-azonosítási rend…

. 

2. ábra: A kódírás folyamatában fellépő támogatások formái (forrás: saját ábrázolás)

1. Ha a kiképzendő számára a fenti (fórum-jellegű gyors kijózanodást, masszív helyzet-átértelmezést lehetővé tevő) segítség precízebb: vö. Sheets("Munka1").Cells(9, 3).Value = "Helló világ" – akkor immár a FORMULA és a VALUE szavak ismeretében felmerülhet benne az első szuverén gondolat ideális esetben: Máris képes lehetek hivatkozást/függvényt létrehozni macro-ból?
   1. ha a már eddig beírt Hello-Vilag-ok közül egyet át szeretne hivatkozni egy másik cellába, akkor a macro-rögzítő az alábbi „segítséget” adja neki (ha két oszloppal jobbra azonos sorba hozza létre a képletét): Application.CutCopyMode = False, ill. ActiveCell.FormulaR1C1 = "=RC[-2]"
   2. ha a kiképzendő további variánsokat szeretne látni, akkor adott cella tartalmát ismét csak két oszloppal jobbra, de azonos sorba írva az alábbiak szerint is létrehozatja a macro-rögzítővel: ActiveCell.FormulaR1C1 = "=LEFT(RC[-2],99)", vagyis úgy vesz át cellatartalmat, hogy veszi a cellának a tartalom hosszát meghaladó karakterszámú részét, vagyis az egész hello-vilag-stringet…
   3. ebben a pillanatban a függvények és a vezérelhető argumentumaik relatív világa derengeni kezd
   4. már csak az nem világos, miként is lehet pl. az egyszer leírt hello-vilag string-et pl. 10-szer egymás után átlósan (azaz egy-egy sorral lejjebb és egy-egy oszloppal jobbra tolva) is behivatkozni?
2. Így jut el a kiképzendő élete első macro-ként értelmezhető ciklusáig, a ciklus igényéig és ennek valóra váltásáig – más szavakkal: így talál rá az online tananyagok között a for-to-next logikára (vö. pl. <https://www.youtube.com/watch?v=IXIuPshtdl8>):
   1. zavaró tényezőként előbb-utóbb belefut a DIM szükségességének, elhanyagolhatóságának kérdéskörébe
   2. előbb-utóbb megérti, hogy
      1. egy logikai egység a macro-ban nem csak egyetlen egy sor lehet, hanem a for és a next között tetszőlegesen sok sornyi utasítás helyezhető el
      2. a next i lényegében azonos azzal a gondolattal, hogy növeljük meg i értékét eggyel és ugorjunk vissza a for-ciklus elejére
      3. szüksége lehet arra, hogy a ciklusváltozó ne csak monoton eggyel legyen növelhető (step)
      4. sőt, hogy a ciklusváltozónak egyáltalán nem kell egyenletes lépésközökben változnia, hanem az ugrásszerűen is vezérelhető (pl. ha megtalálásra kerül viszonylag gyorsan a cikluson belül az, amiért a ciklus végéig is kellhetett volna várni az eredményre)
      5. a ciklusváltozók kiválóan alkalmasak (mert pl. ezek értéke már elege az induló és a záró érték kapcsán szabadon definiálható előre – és/vagy eleve műveletekkel tetszőleges jelleggel transzformálható) az intelligens kockás füzetben való navigálásra, vagyis a képletek paramétereinek, a függvények argumentumainak dinamikus tartalommal való ellátására, …
3. Az oktató videók kapcsán, ha figyelmes a kiképzendő, azaz követi az alapvető jótanácsot, miszerint kódírás helyett előbb érdemes a kódolvasást gyakorolni, bele fog futni azokba a menekülési stratégiákra utaló jelekbe, melyek segítségével ott is vezérelhető paramétereket tud teremteni, ahol az addig nem tűnt megoldhatónak (vö. 3. ábra)
   1. vagyis megérti az & jel speciális szerepét
   2. a zárójelek és idézőjelek logikáját
   3. ill. azt, hogy trial&error jelleggel lehet kísérlezetni, vajon milyen macro-beállításból mi következik az eredménycellákban, ill. mikor hol lesz az eredménycella maga…



3. ábra: Az eddig merevnek hitt paraméterek/argumentumok relativálása

# Konklúziók

Ezen tapasztalatok birtokában, melyekre egyedül nem triviális rájönni, s melyek lényegében egyetlen egy feladatban (<https://miau.my-x.hu/digeco/coco_prg1.xlsm>) már mind-mind tálcán felkínálásra kerültek, úgy kezdhető meg a macro-írás saját hatáskörben, hogy elvileg a kiképzendő már olyan rugalmasan bevethető eszköztárral rendelkezik (elvileg), mely alig/sehogyan sem korlátozza komplex feladatok megoldásában.

Természetesen a maximális figyelem, koncentráció, motiváltság és az állandó, minden részeredményre kiterjedő önellenőrzés igénye és gyakorlata ekkor sem spórolható meg, mert minden egyéb slamposság meredeken vezet szerencsés esetben a rossz kódhoz, peches esetben téves, de működő kódhoz…

Az így elérhető szint saját szükségletek kielégítésére már alkalmas, de minőségi macro-k nem feltétlenül hozhatók létre – vagyis a macro-írásnak, mint minden programozási feladatnak eltérő komplexitású szintje vannak (vö. <https://miau.my-x.hu/miau/253/cipher1-2-3.docx>).