

Outlook vezérlése Excellel

Az alábbi bemutató célja, hogy egy bevezető tudást adjon a felhasználónak, hogyan tudja könnyedén és egyszerűen irányítani és utasítani az Excel segítségével az Outlook-ot, hogyan tudja összekapcsolni a kettőt, és felhasználni ezt a mindennapi életben vagy a munka világában. Először egy szimpla email elküldésén keresztül bemutatom, hogyan kell a két applikációt összekapcsolni, majd az email szerkezeti paramétereinek dinamizálásáról lesz szó, végül pedig egy a gyakorlatban is igen hasznos dologgal zárom az ismertetőt, mégpedig azzal, hogy hogyan lehet cellatartományokat a levél törzsébe illeszteni, és ezt az egész folyamatot dinamizálni.

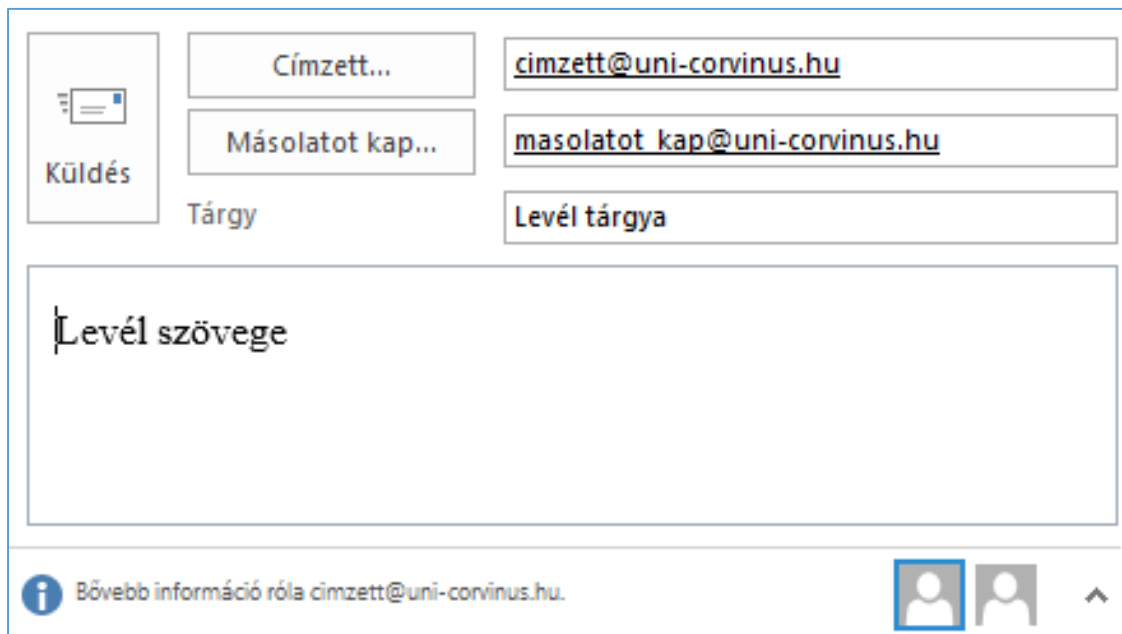
1. Szimpla email küldése Excel-lel Outlook-ból

```
Sub simple_mail()  
  
    Dim OutApp As Object  
    Dim OutMail As Object  
  
    On Error GoTo Clean_up          'hibakezelés - fontos!  
  
    With Application  
        .DisplayAlerts = False  
        .EnableEvents = False  
        .ScreenUpdating = False  
    End With  
  
    Set OutApp = CreateObject("Outlook.Application")  
    Set OutMail = OutApp.Createitem(0)  
  
    With OutMail  
        .To = "cimzett@uni-corvinus.hu"  
        .Cc = "masolatot_kap@uni-corvinus.hu"  
        .Subject = "Levél tárgya"  
        .HTMLBody = "Levél szövege"  
        .Display          'vagy .Send is akár  
    End With  
  
Clean_up:  
    Set OutApp = Nothing  
    Set OutMail = Nothing  
  
    With Application  
        .DisplayAlerts = True  
        .EnableEvents = True  
        .ScreenUpdating = True  
    End With  
  
End Sub
```

A makró láthatóan relatív rövidke, azonban minden sora kifejezett fontossággal bír. Rögtön az elején a két később felhasználandó objektum típusú változó dimenzionálása után egy hibakezelő sorral kezdődik, melynek relevanciáját egy pár sorral lentebb részletezem. Ezután csupán kényelmi szempontból ajánlatos a makró futásáig a DisplayAlerts (hibaüzenetek, felhasználónak címzett comboboxok, inputboxok és messageboxok felbukkanása), az EnableEvents (különböző eseményekre reagáló makrók lefutása) és a ScreenUpdating (a makró által csinált lépések azonnali vizualizálása) kikapcsolása.

Ezt követően megtörténik a két létrehozott objektum típusú változó meghatározása. Az „OutApp” objektumot lényegében a CreateObject() beépített függvényen keresztül maga az Outlook applikációvá tesszük. Az „OutMail” pedig a már létrehozott OutApp, azaz az Outlook mail típusú objektuma lesz. Ezt a CreateItem() metódussal érjük el, melyben a 0 érték a mail típust determinálja.

Tehát lényegében az előbb két lépésben létrehoztunk egy üres emailt, melyet most tartalommal kellene megtölteni. Ehhez fogjuk az „OutMail” (létrehozott üres email) objektumunkat és a viszonylag beszédes nevű paramétereinek értékeket adunk. A kódban megadott paraméterek alapján a következő emailt fogjuk viszont látni a futtatás eredményeképp:



Fontos, hogy ajánlatos a .Display parancs használata, ugyanis ezzel az elküldés előtti formátumban megkapjuk a levelet, így még van lehetőség egyszer gyorsan átfutni, a bekerült hibákat esetleg manuálisan javítani, majd a Küldés gombra kattintva az útjára eresztetni. A bátrabbak persze

nyugodtan használhatják a .Send parancsot is, mellyel értelemszerűen előnézet nélkül rögtön elküldésre kerül az email.

Végül pedig a hibakezelős sorra visszatérve és így a makró végét is magyarázva: ez a rész arra szolgál, hogy a makró végén látható (a kód bejezéseképp amúgy is lefutó) „Clean_up” részre ugorjon, melynek feladata először is a két objektum típusú változót megszüntetése, majd a fentebb említett kényelmi okokból kikapcsolt 3 funkciót visszakapcsolása az Excel további alapértelmezett működéséhez. Az objektumok megszüntetése azért nagyon fontos minden egyes, akár sikeres, akár sikertelen futás után, mivel ezek megszüntetés nélkül továbbra is léteznének. Így egyrészt foglalnak a helyet a memóriában, valamint későbbi futásoknál „össze is akadhatnak” az akkor létrehozott objektumok a már létezőkkel (ez utóbbi probléma inkább bonyolultabb kódoknál és más applikációknál pl. PowerPoint jellemzőbb).

2. Paraméterek dinamizálása

Ebben a pontban koncentráljunk a fentebbi makró lényegi részére, azaz amikor az üres email objektumot különböző értékekkel láttuk el. Ezt az imént úgynevezett „hard-coded” (statikus) értékekkel történt, azaz minden paraméterhez egy fix (itt minden esetben szöveg típusú) értéket rendeltünk hozzá. Vegyük észre azonban, hogy ezeket könnyedén dinamizálni tudjuk, és így iterálhatóvá tehető az egész folyamat.

Nézzük meg ezt egy egyszerű, gyakorlatias, hétköznapi példán keresztül. Tegyük fel, hogy SZPM-t tanítunk egy 5 fős csoportnak, akiről az alábbi adatbázis áll rendelkezésünkre:

	A	B	C	D	E	F
1	Név	Email cím	Szak		Idő adatok képlettel	
2	A. András	a.andras@uni-corvinus.hu	pénzügy		Ma	Hónap
3	B. Béla	b.bela@uni-corvinus.hu	számvitel		2018.11.02	november
4	C. Cecília	c.cecilia@uni-corvinus.hu	pénzügy			
5	D. Dénes	d.denes@uni-corvinus.hu	pénzügy			
6	E. Emília	e.emilia@uni-corvinus.hu	számvitel			

Írnunk kéne nekik, hogy az e havi óráknál teremváltkozás lesz, viszont a Neptun szokás szerint lehalt, és szeretjük a személyeskedést, ezért mindenkinek külön íránk erről. Ehhez az alábbi kódot raktuk össze a fenti kód kiegészítésével és helyenkénti megváltoztatásával (a 3 db „kényelmi funkció” be- és kikapcsolásától most eltekinthetünk):

```

Sub more_mail()

    Dim OutApp As Object
    Dim OutMail As Object
    Dim i, num_stud As Integer
    Dim month_name, name, email_ad As String

    On Error GoTo Clean_up          'hibakezelés - fontos!

    Set OutApp = CreateObject("Outlook.Application")

    num_stud = 5                    '5 db diák
    month_name = Cells(3, 6)        'most október

    For i = 2 To num_stud + 1
        name = Cells(i, 1)
        email_ad = Cells(i, 2)

        Set OutMail = OutApp.Createitem(0)

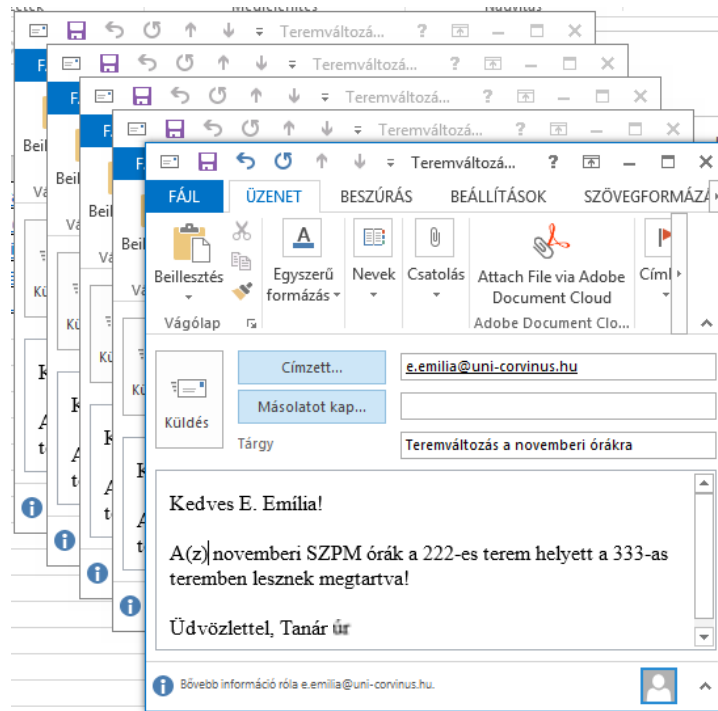
        With OutMail
            .To = email_ad           'email cím iterálása
            .Subject = "Teremváltás a " & month_name & "i órákra" 'hónap változóként
            .HTMLBody = "Kedves " & name & "!" & _
                "<br>" & "<br>" & _
                "A(z) " & month_name & "i SZPM órák a 222-es terem helyett " & _
                "a 333-as teremben lesznek megtartva!" & _
                "<br>" & "<br>" & _
                "Üdvözlettel, Tanár úr"           'név (megszólítás) iterálása
            .Display                     'vagy .Send is akár
        End With
    Next i

Clean_up:
    Set OutApp = Nothing
    Set OutMail = Nothing

End Sub

```

Látható, hogy a kód kiegészült egy pár dinamikus változóval, mint például a név (name) és az email cím (email_ad) változók, melyek iterációnként új értéket kapnak (mindig a soron levő diák adatait), valamint belecsempésztük a hónap nevét is egy változóként, melyet ugyancsak az excel munkalapról kap fel a makró. A címzettek közti iterációt egy paraméterezett for ciklus biztosítja. A levél szövegében még feltűnhet egy új dolog, a
 tag, mely a szövegek közti sortörést jelenti (a dupla
 pedig tehát egy sor kihagyását, amit persze máshogy is lehetne csinálni). A makrót lefuttatva a várt 5 emailt kapjuk eredményül, így már csak gyorsan át kell futnunk őket, és máris küldhetőek egy-egy gombnyomással:



Tehát sikerült összerakni egy olyan munkalapot, mellyel a hónap nevének átírásával és a címzett lista karbantartásával, egy gombnyomással képesek vagyunk előre megírt, sorozatos emaileket küldeni. Mivel a fenti példából is látszik, hogy a levél bármely része könnyedén dinamizálható, ez a technika rendkívül hasznos tud lenni hasonló esetekben.

3. Címzett-listák menedzselése

Feltűnhetett, hogy a fenti példában eddig a „Szak” oszlopot nem használtuk. Ezt azért raktam bele, hogy egy kicsit a címzett listák alapvető kezelésében is elmélyedhessünk. Tegyük fel, hogy a teremváltás csak a pénzügyes hallgatókat érinti. Ha a fenti makrót egy if-fel kiegészítjük az alábbi módon, máris egy alapszintű címzettlista-menedzsmentet sajátítottunk el és építettünk bele kódunkba:

```

For i = 2 To num_stud + 1
  If Cells(i, 3) = "pénzügy" Then 'címtettlista-menedzsmment!
    name = Cells(i, 1)
    email_ad = Cells(i, 2)

    Set OutMail = OutApp.Createitem(0)

    With OutMail
      .To = email_ad 'email cím iterálása
      .Subject = "Teremváltás a " & month_name & "i órákra" 'hónap változóként
      .HTMLBody = "Kedves " & name & "!" & _
        "<br>" & "<br>" & _
        "A(z) " & month_name & "i SZPM órák a 222-es terem helyett " & _
        "a 333-as teremben lesznek megtartva!" & _
        "<br>" & "<br>" & _
        "Üdvözlettel, Tanár úr" 'név (megszólítás) iterálása
      .Display 'vagy .Send is akár
    End With
  End If
Next i

```

Ezt lefuttatva ugyanis már csak a 3 pénzügyesünk kapja meg a szánt emailt, és mindehhez csupán az if – end if klauzulát kellett beékelni. Az Outlook-nak egyébként ennél jóval kifinomultabb és egyszerűbb címtett-lista kreálási és menedzselési opciói vannak, de az alábbi problémára tökéletes és gyors megoldást nyújtott az Excel-es módszer is. További előnye még az Excelnek, hogy a besorolásokat szabadon bővíthetjük, egy email/személy többfajta besorolási oszlop segítségével több különböző csoportba is tartozhat, így könnyedén „filterezhetővé” tehető egy-egy nagyobb címtett-adatbázis.

4. Excel-tartományok küldése

Végül szeretném bemutatni az egyik leginkább hasznos funkciót, amit ebben a témában el lehet sajátítani, mégpedig azt, hogyan lehet range-eket küldeni egy levélben, és természetesen mindezt automatizálni. Ahogy a VBA-ban (és úgy általában a programozás területén) mindent, ezt is sokféleképpen lehet megoldani. A bemutatásra váró módszert azért ajánlom és én is azért használom előszeretettel, mert az adott range-t az excelben már előre megformázhatom és tárolhatom, és ugyanazt ugyanúgy (formázott range-ként, nem pedig pl.: szimpla keretek nélküli range-ként vagy képként) fogom viszontlátni a leveleimben is.

Mindezt úgy érem el, hogy az adott választott excel range-t egy közepesen bonyolult folyamaton keresztül egy htm (html) típusú fájlban kimentem, majd ebből a fájlból egy változóba visszaolvasva, majd ezt a levél objektumnak „átadva” kapom meg a kívánt eredményt. Alább látható a függvény, mely ezt a folyamatot véghezviszi. A függvény alapvetően a netről van (<https://www.rondebruin.nl/win/s1/outlook/bmail2.htm> vagy google-be beírva is megtalálható a „range to html vba” első keresési eredményeként), már évek óta használom. A bonyolultsága miatt

a kód minden sorát igyekeztem felkommentelni, valamint még alább is végigmegyek a lépéseim majd egyszer. Íme a sokat emlegetett function:

```
Function range_to_html(rng As Range)

    Dim fso As Object
    Dim ts As Object
    Dim TempFile As String
    Dim TempWB As Workbook

    'az Environ fv lekéri a temp mappa elérési útját; ez általában: "C:\Users\<username>\AppData\Local\temp"
    'majd ide készít egy üres html fájlt melynek neve a pontos idő lesz (esetleges megkülönböztetés miatt!)
    TempFile = Environ$("temp") & "\" & Format(Now, "dd-mm-yy h-mm-ss") & ".htm"

    rng.Copy          'a függvény inputjaként megadott range-t másolja
    Set TempWB = Workbooks.Add(1)          'létrehoz egy új átmeneti munkafüzetet
    With TempWB.Sheets(1)
        .Cells(1).PasteSpecial Paste:=8
        .Cells(1).PasteSpecial xlPasteValues, , False, False
        .Cells(1).PasteSpecial xlPasteFormats, , False, False
        .Cells(1).Select          'beilleszti az előbb másolt range-t az A1-es cellából kezdve
        Application.CutCopyMode = False          'vágólapról törli a tartalmat
        On Error Resume Next
        .DrawingObjects.Visible = True          'ha esetleg volt bármilyen alakzat/ábra a másolt range-n, megjeleníti ...
        .DrawingObjects.Delete          '... majd törli
        On Error GoTo 0
    End With

    'Kiment a kreált munkafüzet használt munkalapjáról a használt tartományt egy html fájlba
    With TempWB.PublishObjects.Add( _
        Source:=TempWB.Sheets(1).UsedRange.Address, _
        Filename:=TempFile, _
        Sheet:=TempWB.Sheets(1).Name, _
        Source:=TempWB.Sheets(1).UsedRange.Address, _
        HtmlType:=xlHtmlStatic)
        .Publish (True)          'publish = "mentés"
    End With

    'a range_to_html-be beolvassa az imént kimentett html fájl tartalmát
    Set fso = CreateObject("Scripting.FileSystemObject")          'ezzel az objektummal lehet különféle fájltípusokat kezelni
    Set ts = fso.GetFile(TempFile).OpenAsTextStream(1, -2)          '1 -> csak olvasásra; -2 -> alapértelmezett programmal
    range_to_html = ts.ReadAll
    ts.Close
    range_to_html = Replace(range_to_html, "align=center x:publishsource=", _
        "align=left x:publishsource=")          'alapértelmezett középre zárt helyett balra zárt (esztétika!)

    TempWB.Close savechanges:=False          'használt átmeneti excelből mentés nélküli kilépés

    Kill TempFile          'használt átmeneti htm fájl törlése
    Set ts = Nothing
    Set fso = Nothing
    Set TempWB = Nothing

End Function
```

Tehát a részletes kommenteket kiegészítvén a kód menete még egyszer összefoglalva:

1. Egy üres htm fájl készítése egy alapértelmezett ideiglenes fájlokat tartalmazó mappába, hogy majd ebbe tudjuk kimenteni a választott range-t
2. Kimásoljuk az alap munkafüzetünkben a választott range-t, majd egy új excel munkafüzetet hozunk létre, melybe beillesztjük, majd a folyamat során esetleg feleslegesen átmásolt objektumoktól (alakzatok, szövegdobozok, stb...) megtisztítjuk
3. A kreált munkafüzetből kimentjük a használt tartományt az 1. pontban készített üres htm fájlba

4. A függvényünk kimeneti értékébe olvassuk az imént kimentett htm fájl tartalmát, azaz a kívánt range-t
5. Mentés nélkül bezárjuk a 2. pontban létrehozott ideiglenes munkafüzetet (ezáltal „töröljük”/elvetjük, mivel nem volt mentve), majd töröljük a htm fájlt, és végül megsemmisítjük a függvény során készített és használt objektumainkat

Most, hogy megvan, hogyan tudunk egyszerűen és a beállított formázást tartva range-t másolni, nézzük meg hogyan lehetne ezt alkalmazni az előző példákon keresztül. Tegyük fel, hogy szeretnénk kiküldeni mindenkinek az eddigi zh pontjait, amihez az adatokat egy hasonló excel-ben tároljuk:

	A	B	C	D	E	F
1	Név	Email cím	Szak	1. zh pont	2. zh pont	3. zh pont
2	A. András	a.andras@uni-corvinus.hu	pénzügy	5	4	14
3	B. Béla	b.bela@uni-corvinus.hu	számvitel	6	7	13
4	C. Cecília	c.cecilia@uni-corvinus.hu	pénzügy	10	8	5
5	D. Dénes	d.denes@uni-corvinus.hu	pénzügy	12	2	6
6	E. Emília	e.emilia@uni-corvinus.hu	számvitel	0	1	10

A megírt leveinkbe a zh-pontokat tartalmazó cellákat rendre nevenként be szeretnénk majd másolni range-ként. Ehhez egy picit változtatunk az előző kódok struktúráján, ugyanis valahol meg kell adnunk az email paraméterei mellett a kívánt range-eket is. Ezt persze, ahogy már fentebb is említettem, először is a makróban tesszük meg a szokásos set paranccsal (a range objektum megadása), majd ezt a fenti range_to_html függvényrel (amit én a levél „body”-jába tettem) beolvasztatjuk és küldhetővé alakítjuk. Erre az alábbi kódot fogjuk alkalmazni, a lényegesen eltérő és fentebb külön kifejtett sorokat piros ponttal emeltem ki:


```

Sub more_mail_wrangle()
    Dim OutApp As Object
    Dim OutMail As Object
    Dim i, num_stud As Integer
    Dim month_name, name, email_ad As String
    Dim rng As Range           'újdonsült range változó

    On Error GoTo Clean_up     'hibakezelés - fontos!

    Set OutApp = CreateObject("Outlook.Application")

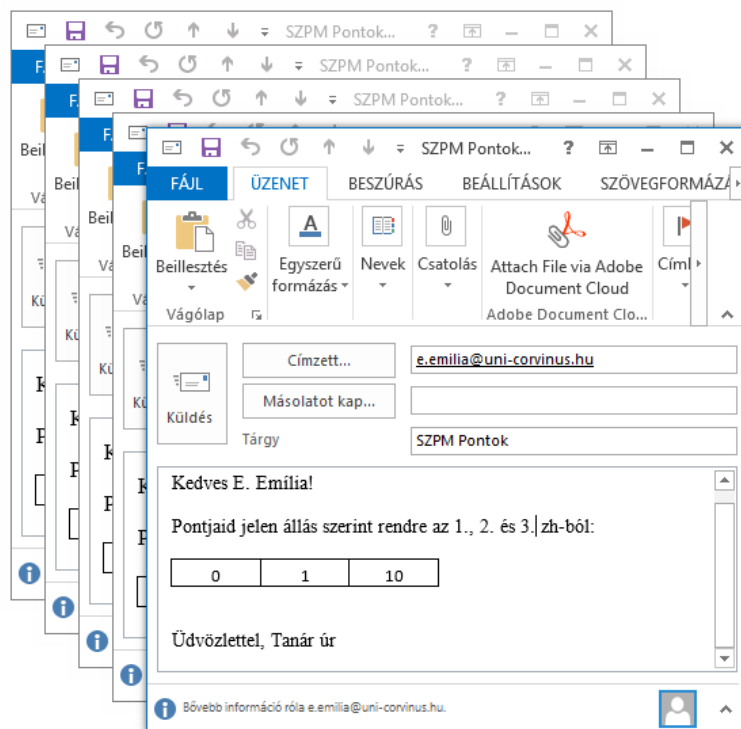
    num_stud = 5                '5 db diák
    month_name = Cells(3, 6)    'most október

    For i = 2 To num_stud + 1
        name = Cells(i, 1)
        email_ad = Cells(i, 2)
        Set rng = Range(Cells(i, 4), Cells(i, 6))    'cellák amikben a pontok vannak
        Set OutMail = OutApp.CreateItem(0)
        With OutMail
            .To = email_ad           'email cím iterálása
            .Subject = "SZPM Pontok"
            .HTMLBody = "Kedves " & name & "!" & _
                "<br>" & "<br>" & _
                "Pontjaid jelen állás szerint rendre az 1., 2. és 3. zh-ból:" & _
                range_to_html(rng) & _
                "<br>" & _
                "Üdvözlettel, Tanár úr"
            'név (megszólítás) iterálása és pontok range (rng)!!!
            'vagy .Send is akár
            .Display
        End With
    Next i

Clean_up:
    Set OutApp = Nothing
    Set OutMail = Nothing
End Sub

```

A kód lefuttatásával a várt emaileket kapjuk eredményül, melyek már csak küldésre várnak:



Összefoglalás

A fenti dokumentum tehát támpontot nyújt, hogyan lehet az excel-ből az outlook-ot vezérelve maileket küldeni. Ehhez bemutatásra került, hogyan lehet maileket küldeni statikus, majd dinamikus paraméterezéssel, valamint hogyan lehet alapvető címzett-listákat menedzselni, és végül hogy hogyan tudunk range-eket az emailjeinkbe illeszteni. A felhasznált kódokat közben csak képeken mutattam be, de a lenti mellékletben másolható formában is megosztom őket, hátha valakinek kedve támad kipróbálni vagy akár kisebb módosításokkal rendszeres használatba állítani őket.

Melléklet

A mellékelte makrók:

- simple_mail sub: email küldése egy címzettnek statikus paraméterekkel (1. pont)
- more_mail sub: több címzettnek mail küldése dinamikus paraméterekkel (2. pont) – ezt lehet még kiegészíteni if-end if klauzulákkal címzett listák menedzseléséhez (3. pont)
- more_mail_wrange sub: range-eket tartalmazó mailek küldése (4.pont)
- range_to_html function: a range-eket átalakító és bemásoló függvény (4.pont)

Option Explicit

Sub simple_mail()

Dim OutApp As Object
Dim OutMail As Object

On Error GoTo Clean_up 'hibakezelés - fontos!

With Application

.DisplayAlerts = False
.EnableEvents = False
.ScreenUpdating = False

End With

Set OutApp = CreateObject("Outlook.Application")
Set OutMail = OutApp.Createitem(0)

With OutMail

.To = "cimzett@uni-corvinus.hu"
.Cc = "masolatot_kap@uni-corvinus.hu"
.Subject = "Levél tárgya"
.HTMLBody = "Levél szövege"
.Display 'vagy .Send is akár

End With

Clean_up:

Set OutApp = Nothing
Set OutMail = Nothing

```

With Application
    .DisplayAlerts = True
    .EnableEvents = True
    .ScreenUpdating = True
End With

End Sub

Sub more_mail()

    Dim OutApp As Object
    Dim OutMail As Object
    Dim i, num_stud As Integer
    Dim month_name, name, email_ad As String

    On Error GoTo Clean_up      'hibakezelés - fontos!

    Set OutApp = CreateObject("Outlook.Application")

    num_stud = 5                '5 db diák
    month_name = Cells(3, 6)    'most október

    For i = 2 To num_stud + 1
        name = Cells(i, 1)
        email_ad = Cells(i, 2)

        Set OutMail = OutApp.Createitem(0)

        With OutMail
            .To = email_ad                'email cím iterálása
            .Subject = "Teremváltás a " & month_name & "i órákra" 'hónap változóként
            .HTMLBody = "Kedves " & name & "! " & _
                "<br>" & "<br>" & _
                "A(z) " & month_name & "i SZPM órák a 222-es terem helyett " & _
                "a 333-as teremben lesznek megtartva!" & _
                "<br>" & "<br>" & _
                "Üdvözlettel, Tanár úr"      'név (megszólítás) iterálása
            .Display                'vagy .Send is akár
        End With
    Next i

Clean_up:
    Set OutApp = Nothing
    Set OutMail = Nothing

End Sub

Sub more_mail_wrange()
    Dim OutApp As Object
    Dim OutMail As Object
    Dim i, num_stud As Integer
    Dim month_name, name, email_ad As String
    Dim rng As Range          'újdonült range változó

    On Error GoTo Clean_up      'hibakezelés - fontos!

    Set OutApp = CreateObject("Outlook.Application")

    num_stud = 5                '5 db diák
    month_name = Cells(3, 6)    'most október

```

```

For i = 2 To num_stud + 1
    name = Cells(i, 1)
    email_ad = Cells(i, 2)
    Set rng = Range(Cells(i, 4), Cells(i, 6))    'cellák amikben a pontok vannak
    Set OutMail = OutApp.Createitem(0)
    With OutMail
        .To = email_ad                        'email cím iterálása
        .Subject = "SZPM Pontok"
        .HTMLBody = "Kedves " & name & "! " & _
            "<br>" & "<br>" & _
            "Pontjaid jelen állás szerint rendre az 1., 2. és 3. zh-ból:" & _
            range_to_html(rng) & _
            "<br>" & _
            "Üdvözlettel, Tanár úr"
            'név (megszólítás) iterálása és pontok range (rng)!!!
    .Display                                'vagy .Send is akár
    End With
Next i

Clean_up:
    Set OutApp = Nothing
    Set OutMail = Nothing
End Sub

Function range_to_html(rng As Range)

    Dim fso As Object
    Dim ts As Object
    Dim TempFile As String
    Dim TempWB As Workbook

    'az Environ fv lekéri a temp mappa elérési útját; ez általában: "C:\Users\<username>\AppData\Local\temp"
    'majd ide készít egy üres html fájlt melynek neve a pontos idő lesz (esetleges megkülönböztetés miatt!)
    TempFile = Environ$("temp") & "\" & Format(Now, "dd-mm-yy h-mm-ss") & ".htm"

    rng.Copy                                'a függvény inputjaként megadott range-t másolja
    Set TempWB = Workbooks.Add(1)           'létrehoz egy új átmeneti munkafüzetet
    With TempWB.Sheets(1)
        .Cells(1).PasteSpecial Paste:=8
        .Cells(1).PasteSpecial xlPasteValues, , False, False
        .Cells(1).PasteSpecial xlPasteFormats, , False, False
        .Cells(1).Select                    'beilleszti az előbb másolt range-t az A1-es cellából kezdve
        Application.CutCopyMode = False    'vágólapról törli a tartalmat
    On Error Resume Next
        .DrawingObjects.Visible = True     'ha esetleg volt bármi alakzat/ábra a másolt range-n, megjeleníti ...
        .DrawingObjects.Delete             '... majd törli
    On Error GoTo 0
    End With

    'Kimentti a kreált munkafüzet használt munkalapjáról a használt tartományt egy html fájlba
    With TempWB.PublishObjects.Add( _
        SourceType:=xlSourceRange, _
        Filename:=TempFile, _
        Sheet:=TempWB.Sheets(1).name, _
        Source:=TempWB.Sheets(1).UsedRange.Address, _
        HtmlType:=xlHtmlStatic)
        .Publish (True)                    'publish = "mentés"
    End With

    'a range_to_html-be beolvassa az imént kimentett html fájl tartalmát
    Set fso = CreateObject("Scripting.FileSystemObject")    'ezzel az objektummal lehet külső fájltypusokat kezelni
    Set ts = fso.GetFile(TempFile).OpenAsTextStream(1, -2)    '1 -> csak olvasásra; -2 -> alapértelmezett programmal

```

```

range_to_html = ts.readall
ts.Close
range_to_html = Replace(range_to_html, "align=center x:publishsource=", _
    "align=left x:publishsource=") 'alapértelmezett középre zárt helyett balra zárt (esztétika!)

TempWB.Close savechanges:=False 'használt átmeneti excelből mentés nélküli kilépés

Kill TempFile 'használt átmeneti htm fájl törlése
Set ts = Nothing
Set fso = Nothing
Set TempWB = Nothing

End Function

```