Az OpenOffice.org programozása

Egy elveszett fejlesztőkörnyezet titkai kedvenc irodai csomagunkban.

Az *OpenOffice.org* révén a *Linux* mára egyértelműen eljutott arra a szintre, hogy teljes értékű helyettesítője lehet egy windowsos irodai munkaállomásnak. Mit is jelent ez? Tartalmaz egy szövegszerkesztőt, egy táblázatkezelőt, van benne bemutató-készítő, egy rajzoló-program, sőt mi több, egy remekbe szabott képletszerkesztőt is találunk a csomagban. Mindezt egy ízléses felhasználói felületen, és nem utolsó sorban széles fájlformátum támogatással.

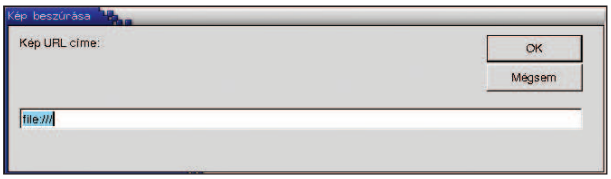
Mindenünk megvan? Mint tudjuk, az *MS Office* tartalmaz egy önálló adatbázis kezelőt, *Access* néven, itt mintha ez hiányozna. Ám a súgó rövid tanulmányozása után azonnal kitűnik, hogy nem erről van szó. Az *Eszközök* menü *Adatforrások...* menüpontja alatt beállíthatjuk, hogy mely *Adabas*, *MySQL*, *dBase*, tetszőleges *ODBC*, vagy *JDBC* adatforrást szeretnénk használni. Természetesen a sima szöveges állomány, illetve a munkafüzet is használható.

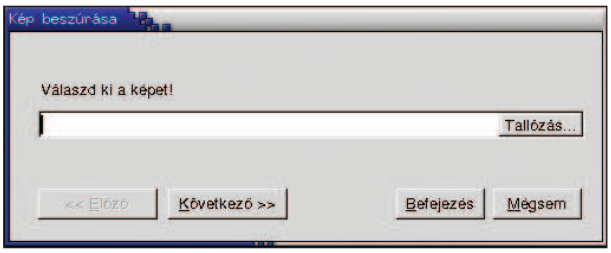
Mintha még mindig nem lenne teljes a kép. Gondolkozzunk csak egy picit. Hát persze, a makrók, és az ezekkel járó vírusok. Ez az, aminek a hiánya eddig álmatlan éjszakákat okozott mindannyiunknak, hát végre *Linux* alatt is elérhetőek! Na jó. Hosszú és parttalan vitát indítana el, ha azt kezdeném el boncolgatni, hogy mi értelme egy közel teljes értékű fejlesztőkörnyezetet építeni egy szövegszerkesztőbe. Inkább nézzük meg, mi mit nyerhetünk abból, ha tudjuk ezt használni.

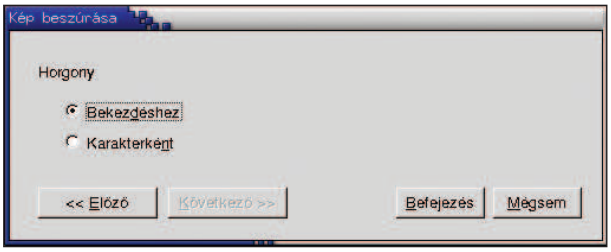
Essünk neki! Nyissunk meg egy új szöveges dokumentumot bármely *OpenOffice.org* alkalmazásból a *Fájl*/*Új*/*Szöveges dokumentum* menüponttal. Azonnal mentsük is el egy beszédes nevű állományba a Fájl/Mentés másként... menüpont segítségével. Legyen mondjuk ez az *ElsoMakrom.sxw*. Ezután válasszuk ki az *Eszközök*/*Makrók*/*Makró...* menüpontot és időzzünk el egy pillanatra.

A most látható ablakban találjuk az elérhető makrók listáját, gyűjtőkbe szervezve. A bal oldali fa struktúra tartalmazza az előbb említett gyűjtőket, azaz azokat a helyeket, ahol *Basic* nyelven írt makrókat találhatunk. Nem említettem volna? Az *OpenOffice.org*-ban is egyfajta objektum-orientált *Basic* nyelven dolgozhatunk, akárcsak a másik irodai csomagban. Visszatérve a gyűjtőkre, egy *soffice* nevűt mindenképp látni fogunk, az ebben található modulok makróit bármely dokumentum elérheti. Ezen kívül minden megnyitott dokumentum egy további gyűjtőt képez.

Dolgozzunk tehát a saját állományunkba. Válasszuk ki a *Makró* forrásaként az *ElsoMakrom.sxw* gyűjtő alatti *Standard könyvtárat*, és kattintsunk az *Új* gombra. Ekkor az *OpenOffice.org* megkér, hogy határozzuk meg a modul nevét. Fogadjuk el nyugodt szívvel a *Module1* nevet, és kattintsunk az *Ok* gombra. Ekkor elindul a *Basic IDE (Integrated Development Environment - Integrált Fejlesztői Környezet)*, melyben két üres eljárás fogad minket.







Mielőtt továbblépnénk, vessünk még egy pillantást a makrók szervezése mögötti elképzelésre. A gyűjtő egy logikai szervező egység, amely egy-egy dokumentumot jelképez, vagy adott esetben maga az *soffice*, amely bárki számára elérhető. A gyűjtők alatt található könyvtárak olyan egységek, melyek moduloknak adhatnak helyet. A modulok jelentik a legkisebb egységet. Egy modul egy forrásállományt jelent, amely természetszerűleg eljárásokból épül fel. Utóbbi eljárások a makrók.

Ezek szerint az előbb látott két üres eljárás, a *Main* és a *Macro1* egyből két új makrót jelent? A válasz igen. Zárjuk be a *Basic IDE*-t és keressük meg újra a *Makró...* menüpontot. Kiválasztva a dokumentumunk *Standard* könyvtárának *Module1* modulját, látható, hogy az előbb

még üres jobb oldal már két makrót tartalmaz. Válasszuk ki bármelyiket és kattintsunk a Szerkesztés gombon. Így ismételten a fejlesztőkörnyezetben találjuk magunkat. A fejlesztőkörnyezet színkiemeléssel segíti a munkát, és hely érzékeny súgó is van, amit az ***F1*** billentyűvel csalogathatunk elő. Sajnos ez utóbbi angol nyelvű, viszont annál bőségesebb. Lássunk végre egy példát egy valódi makróra! Először is töröljük a ***Macro1*** eljárást, majd a Main rutint az alábbiaknak megfelelően írjuk meg:

```
Sub Main

   kepURL = InputBox ("Kép URL címe:", "Kép
   ➥beszúrása", "file:///")
   if kepURL = "" then ' a felhasznalo Megse-t
   ➥nyomott
      exit sub
   endif

   dokumentum = ThisComponent
   szoveg = dokumentum.getText ()

   grafika = dokumentum.createInstance
   ➥("com.sun.star.text.GraphicObject")
   grafika.GraphicURL = kepURL

   lathatoKurzor = dokumentum.getCurrentController
   ➥().getViewCursor ()
   szovegKurzor = szoveg.createTextCursorByRange
   ➥(lathatoKurzor.getStart ())

   szoveg.insertTextContent (szovegKurzor.getStart
   ➥(), grafika, false)

End Sub
```

Ennek az egyszerű makrónak az a feladata, hogy egy képet szúrjon be arra a helyre, ahol a kurzor áll. Megjelenít egy beviteli ablakot, ahol a kép URL-je határozható meg. Az ***Ok*** gombra kattintva a kép beszúrásra kerül. Ennél azonban felhasználóbarátabb megoldást is alkalmazhatunk. Készítsünk egy igazi tündért, és helyettesítsük ezzel a ***Beszúrás*** menü ***Kép*** pontját.
Ehhez már egy önálló párbeszédablakra lesz szükségünk. A ***Basic IDE***-ben alul láthatjuk azt a fület, amely jelzi, hogy épp a ***Module1*** modult szerkesztjük. Kattintsunk ezen a fülön jobb egérgombbal és válasszuk az előugró helyi menüből a ***Beszúrás/BASIC***-párbeszédablak pontot. A párbeszédablak szerkesztőjét látjuk, egyelőre egy üres ablakkal. Ettől kezdve az alul található fülekkel ugrálhatunk a forráskód és az ablak szerkesztője között.
Vezérlők elhelyezéséhez meg kell nyitnunk a ***Vezérlőelemek*** ablakot. Ezt a felül található eszköztárak azon ikonján történő kattintással csalogathatjuk elő, melyen egy jelölőnégyzet, egy választógomb, valamint egy gördítősáv egyszerre látható. Innen kedvünkre válogathatunk a vezérlők között az olyan egyszerűektől, mint a nyomógomb, egészen az olyan összetettekig, mint a fájlkiválasztás.
Rakjunk is fel egy fájlkiválasztó elemet az ablakra. Az elem felhelyezése, átméretezése és mozgatása magától értetődő, játszunk is el vele egy picit. Ha már elégedettek vagyunk a munkánkkal, helyezzünk el négy új gombot is. Bármely elemen a jobb egérgombbal előhívhatjuk a helyi menüt. Itt a ***Tulajdonságok...*** menüpont kiválasztásával, vagy az elemen kettős kattintással megnyithatjuk az elem Tulajdonságok lapját. A gomboknak adhatunk saját nevet, melyre a makrókból hivatkozhatunk, illetve egy szövegcímkét. Első gombunkat nevezzük el `Elozo`-nek, címkéje legyen ***„<< Előző"***. A következő neve legyen `Kovetkezo`, címkéje pedig ***„Következő >>"***. A továbbiaknak legyen `Befejezes` neve, ***Befejezés*** címkéje, illetve `Megsem` neve, ***Mégsem*** címkéje. Az `Elozo` gomb ***Engedélyezett mezőjét*** pedig állítsuk ***Nem*** értékre.
A felhasználóbarátság jegyében adjunk hozzá az ablakhoz még egy egyszerű címkét, és írjuk bele a ***„Válaszd ki a képet!"*** szöveget. A teljes ablaknak is előhívható a Tulajdonságok lapja, így lehet neki címet adni. Végezetül a ***Vezérlők*** eszköztár utolsó ikonjára kattintva bekapcsolhatjuk a ***Tesztüzemmódot***, ezáltal kipróbálhatjuk, hogyan is fog kinézni az ablak, amikor élesben használjuk.
A ***Tesztüzemmódot*** az ESCAPE billentyűvel hagyhatjuk el. Ahhoz, hogy gombjaink értelmes életet éljenek, eljárásokat kell hozzájuk írnunk, majd ezeket az eljárásokat össze kell kötni a gombok megfelelő eseményeivel. Váltsunk tehát vissza a kódszerkesztőre és programozzunk egy kicsit.
A párbeszédablakunk referenciáját modulszinten határozzuk meg, ez már majdnem olyan, mint egy globális változó. A megvalósítás azonban így jelentősen egyszerűsödik.

```
Private parbeszedAblak as Variant

Sub Main

   parbeszedAblak = createUnoDialog
   ➥(DialogLibraries.Standard.Dialog1)
   parbeszedAblak.execute ()

End Sub

Sub ParbeszedAblakMegsem

   parbeszedAblak.endExecute ()

End Sub

Sub ParbeszedAblakBefejezes

   Dim kepURL as String

   parbeszedAblak.endExecute ()

   ' Eleresi utat fogunk kapni, ezt at kell
   ➥alakitani
   kepURL = ConvertToURL
   ➥(parbeszedAblak.Model.FileControl1.Text)

   dokumentum = ThisComponent
   szoveg = dokumentum.getText ()

   grafika = dokumentum.createInstance
   ➥("com.sun.star.text.GraphicObject")
```

```
   grafika.GraphicURL = kepURL

   lathatoKurzor = dokumentum.getCurrentController
   ➥().getViewCursor ()
   szovegKurzor = szoveg.createTextCursorByRange
   ➥(lathatoKurzor.getStart ())

   szoveg.insertTextContent (szovegKurzor.getStart
➥(), grafika, false)

End Sub
```

A régi `Main` eljárás `ParbeszedAblakBefejezes` nevet kapott, és ennek megfelelően módosult. Az új `Main` létrehoz egy példányt az ablakból és elindítja. A `ParbeszedAblakMegsem` mindössze bezárja az ablakot. A két új eljárást a megfelelő gombokra történő kattintás eseményéhez kell rendelni. Ehhez váltsunk vissza a párbeszédablak szerkesztőre és a gombok Tulajdonságok lapjának második fülén az Inicializáláskor mezőt töltsük ki. Ehhez nincs szükség gépelésre, a mellette található gombbal tallózhatunk a makrók, azaz az eljárások között.
A közel kész modul `Main` makróját elindítva előugrik az ablak, és már képet is tudunk beszúrni. Viszont nem tündér a tündér, ha nincs több oldal, amin a ***Következő*** gomb folyamatos nyomkodásával nem lehet ugrálni, így hát még dolgoznunk kell egy kicsit. Ne szaladjunk azonban előre, hiszen már most is használható makróval van dolgunk. Tegyük még kényelmesebbé a használatát azzal, hogy kiteszünk az eszköztárra egy gombot, amivel azonnal indítható. Egyszerűen kattintsunk az eszköztáron jobb gombbal, és válasszuk a ***Testreszabás...*** pontot. Bal oldalon görgessünk az ***ElsoMakrom.sxw BASIC*** makrók elemig és bontsuk ki. Keressük meg a ***Standard*** könyvtár ***Module1*** moduljának `Main` makróját. Ezután keressük meg a jobb oldalon azt a pontot, ami alá szeretnénk beszúrni az új gombot. Ezután kattintsunk a ***Hozzáadás —>*** gombra. Jópofa ikonokhoz kattintsunk az ***Ikonok...*** gombon. Ellenőrizzük, hogy az új gomb melletti jelölőnégyzet be van-e jelölve. Ha minden rendben, kattintsunk az ***Ok***-ra, és élvezzük az egykattintásos makróindítás hihetetlen élményét.
Térjünk vissza tündérünkhöz és adjunk értelmet az `Elozo` és `Kovetkezo` gomboknak. Mint tudjuk, a beszúrt képet elhelyezhetjük egy, a bekezdéshez kötött horgonyhoz képest, illetve beszúrhatjuk úgy is, mintha egy karakter volna. Bízzuk ezt a döntést is a felhasználóra azáltal, hogy a tündér következő oldalán két választógombbal szembesítjük. Az egyikkel bekezdéshez horgonyozhatja a beszúrandó képet, a másikkal karakterként végezheti el a beszúrást.
Ehhez már oldalakra lesz szükségünk. A párbeszédablak szerkesztőben, ha bármely vezérlő Tulajdonságok lapját megnézzük, láthatjuk, hogy van egy ***Oldal (fázis)*** tulajdonsága. Ez jelenleg az összes elemre 0. Ez azt jelenti, hogy az összes oldalon elérhető az adott elem. Amennyiben ez az érték 1, akkor csak az 1. oldalon, ha 2, akkor csak a 2-on, stb. látható. A léptetés megvalósításához gombjaink ezen tulajdonságát 0-án hagyjuk, hogy mindig látszódjanak, a már meglévő címkét és fájlválasztót 1-re állítjuk, és felveszünk a 2. oldalra néhány vezérlőt.
Először is tehát állítsuk 1-re a címke és a fájlkiválasztó ***Oldal (fázis)*** tulajdonságát. Úgy tűnik, ettől még nem változott semmi. Most állítsuk át az egész ablak ***Oldal (fázis)*** tulajdonságát 2-re. Az előbb említett vezérlők eltűnnek, mintha sosem lettek volna ott. Ezután tegyünk fel egy címkét és két választógombot. A választógombok nevei legyenek `Bekezdeshez` és `Karakterkent`. A `Bekezdeshez` választógomb ***Tulajdonságok*** lapján az ***Állapot*** mezőt állítsuk arra, hogy ***Kijelölve***.
Az előbb elkészített vezérlők ***Oldal (fázis)*** tulajdonsága önműködően 2 lett. Ez nekünk jó is, viszont az ablak oldal tulajdonságát vissza kell állítanunk 1-re, hiszen azt szeretnénk, hogy az ablak létrehozásakor ez legyen az, amit a felhasználó meglát. Most újra neki kell esnünk a kódolásnak, ez viszont már nem lesz nehéz. Szükség van az `Elozo` és `Kovetkezo` gombok eljárásaira, és ki kell egészítenünk a `ParbeszedAblakBefejezes` rutint a horgony kezeléséhez. Lássuk tehát a teljes kódot:

```
Private parbeszedAblak as Variant

Sub Main

   parbeszedAblak = createUnoDialog
   ➥(DialogLibraries.Standard.Dialog1)
   parbeszedAblak.execute ()

End Sub

Sub ParbeszedAblakElozo

   parbeszedAblak.Model.Step = 1
   parbeszedAblak.Model.Elozo.Enabled = false
   parbeszedAblak.Model.Kovetkezo.Enabled = true

End Sub

Sub ParbeszedAblakKovetkezo

   parbeszedAblak.Model.Step = 2
   parbeszedAblak.Model.Elozo.Enabled = true
   parbeszedAblak.Model.Kovetkezo.Enabled = false

End Sub

Sub ParbeszedAblakMegsem

   parbeszedAblak.endExecute ()

End Sub

Sub ParbeszedAblakBefejezes

   Dim kepURL as String, horgony as Long

   parbeszedAblak.endExecute ()

   ‘ Eleresi utat fogunk kapni, ezt at kell
   ➥alakitani
   kepURL = ConvertToURL
   ➥(parbeszedAblak.Model.FileControl1.Text)
```

```
' Ha Allapot = Kijelolve, akkor State = 1

if parbeszedAblak.Model.Karakterkent.State = 1
➥then
   horgony = com.sun.star.text.
   ➥TextContentAnchorType.AS_CHARACTER
elseif parbeszedAblak.Model.Bekezdeshez.State = 1
➥then
   horgony = com.sun.star.text.
   ➥TextContentAnchorType.AT_PARAGRAPH
endif

dokumentum = ThisComponent
szoveg = dokumentum.getText ()

grafika = dokumentum.createInstance
➥("com.sun.star.text.GraphicObject")
grafika.GraphicURL = kepURL
grafika.AnchorType = horgony

lathatoKurzor = dokumentum.getCurrentController
➥().getViewCursor ()
szovegKurzor = szoveg.createTextCursorByRange
➥(lathatoKurzor.getStart ())

szoveg.insertTextContent (szovegKurzor.getStart
➥(), grafika, false)

End Sub
```

Természetesen az `Elozo` és `Kovetkezo` gombokhoz megfelelő eljárásokat még a gombok eseménykezelőinek meg kell adni, de ezt a harci feladatot már az Olvasóra bízom. Sikerült tehát közösen létrehoznunk egy tündért, amely egy képet szúr be a dokumentumba, és még a horgonyt is meg tudjuk adni a segítségével. Készítettünk hozzá gyorsindítót is az eszköztárra, így ezt a szenzációs makrót, mely a már beépített funkcióként elérhető képbeszúrást végzi el hibátlanul, hihetetlenül gyorsan érhetjük el. Érdemes megismerni olyan eszközöket is, melyekről elképzelhető, hogy nem sűrűn fogjuk előnyben részesíteni őket más megoldásokkal szemben. Azért el kell ismernünk, igen gyorsan össze lehetett dobni ezt a kis tündért, és a szükséges fejlesztőkörnyezet már telepítve volt a gépre egy irodai csomag formájában. Nem ebben a környezetben fogjuk megírni a világegyenletet megoldó programot, de kisebb feladatokra, különösen a táblázatkezelés területén, használható eszközre találhatunk a ***Basicben***. A ***http://api.openoffice.org/DevelopersGuide/DevelopersGuide.html*** címről elérhető több, mint 1000 oldalas leírás bőséges ismertetővel szolgál a témával kapcsolatban. A fenti példa is innen származik. Akit érdekelnek az ***OpenOffice.org*** programozóknak nyújtott szolgáltatásai, mindenképpen töltse le a leírást. Akár ***Java*** nyelven is készíthetők ugyanis olyan alkalmazások, melyek az ***OpenOffice.org*** lehetőségeit aknázzák ki. Érdemes egy-két pillantást vetni erre is.
Sok örömet a programozáshoz!

Fülöp Balázs