Képek átméretezése – átdolgozva

PCLinuxOS Magazine – 2024. február

Írta: Paul Arnote (parnote)

Ahogy azt gondolhatjátok, a PCLinuxOS Magazine főszerkesztőjeként RENGETEGET dolgozom képfájlokkal. Ezért, a grafikus fájlokkal dolgozva, mindig keresem az „életemet megkönnyítő” megoldásokat. Végül is, miért kell nekem GIMP-et (vagy bármely más grafikus fájlszerkesztőt) megnyitnom csak azért, hogy konvertáljak, vagy átméretezzek egy képet, amikor mindezt sokkal gyorsabban megtehetem bash-szkripttel? Plusz, a saját bash-szkript lehetővé teszi azt több fájlon egyszerre végrehajtani.

Tizenegy hónapja frissítettem a korábbi szkriptemet, amivel a különféle grafikus formátumok között konvertálok. Ennek során „védelemmel” egészítettem ki, hogy megakadályozzam az eredeti fájl felülírását csak úgy, mint biztosítsam, hogy a tömörítés értéke valahol 0 és 100 között legyen. *(A szkriptek képei a magyarra fordított szkriptekről készültek!)*

Sokszor használom a szkriptemet a grafikus formátumok közötti konvertálásra. De még ennél is gyakrabban használok egy másik szkriptet képek átméretezésére. Ugyanúgy, majdnem naponta. Ezért gondoltam, hogy frissíteném ezt a szkriptet is, kiegészítve az eredeti képfájl felülírásának elkerülésével és gondoskodva, hogy a kép tömörítésének értéke 0 és 100 között legyen.

Akár a „többi” saját bash-szkriptemet, ezt is bárki futtathatja parancssorból, vagy használható Thunar felhasználói műveleteként is. Minthogy többnyire kizárólag Xfce-asztalt használok, ez utóbbi lehetőség nagyon fontos nekem. Megkönnyíti az életemet és beleillik abba a munkamenetbe, amivel dolgozni szeretek.

A szkript kiolvassa a fájl(ok) nevét és a paraméterei(ke)t. Lehet egy, vagy akár egy teljes köteg fájl. KDE,- vagy Mate-felhasználók is tudják hasonló módon használni a szkriptet, de az, hogyankell beállítani Dolphin, vagy Caja felhasználói menüelemnek, a cikknek nem témája.

A cikkben egy kicsit később bemutatom a szkript egy verzióját, ami néhány változtatással önálló programként működik. Ez lehetővé teszi azoknak, akik ódzkodnak (vagy kerülik) a parancssort, hogy panelen, vagy az asztalon indítót készítsenek egy teljesen önálló programhoz. Így***BÁRMELY*** Linux-asztal használója a szkriptet anélkül futtathatja, hogy bele kellene ásnia magát a nem-Thunar fájlkezelők beállításának rejtelmeibe. Noha én egyáltalán nem vagyok parancssori „varázsló”, ki tudtam találni, hogyan adjak hozzá fájlválasztó ablakot, hogyan használjam fel az információkat, illetve a képek csoportos kiválasztása és átméretezése módját. NEM kell félni, hogy bármelyik eredeti fájlodat felülírná, mivel ezt a lehetőséget megszüntettem. EGYETLEN esetben fordulhat elő BÁRMELY eredeti fájl felülírása, ha ugyanazokat a méret- és tömörítési adatokat adsz meg, mint ami az eredeti fájlé volt, ekkor felül fogja írni a kép RÉGI verzióját, ugyanazon méret- és tömörítési beállításokkal. Máskülönben az eredeti fájl(ok) védett(ek) a felülírástól.

Az „új” szkript

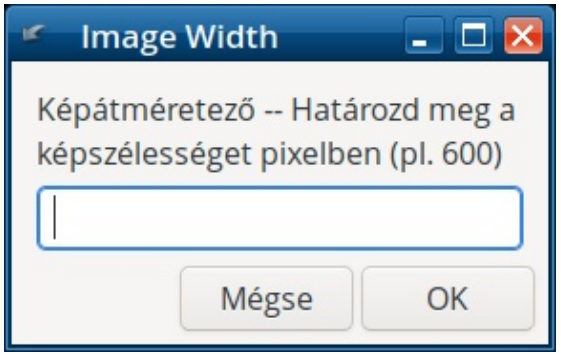
Először a képátméretező szkript „eredeti” verziójától (görgess le annak a cikknek az „Átméretezés” részéig) egy kicsit több mint 10 éve írtam. (Uram Atyám! … Nehéz elhinni, hogy ez OLYAN régen volt!) Az évek során módosítottam, majd ismét módosítottam egy kicsit, és megint csak változtattam rajta egy keveset. Gondolhatod! A szkript új verziója már az **Img-resize4.sh** nevet viseli, A Magazin honlapjáról letöltheted,vagy begépelheted, ha jobban szeretnéd.

Ha a Magazin weboldaláról való letöltés mellett döntesz, figyelj arra, hogy a letöltött fájl „.txt” kiterjesztését eltávolítsd, olyan helyre rakd, ahol a bash-szkriptjeidet tárolni szoktad és végrehajthatóvá tedd a fájlt. Én a bash-szkriptjeimet a ~/bin könyvtárban tartom, ami szerepel a rendszer $PATH kifejezésében. Ezáltal nem kell beírnom a bash-szkriptjeim útvonalát.

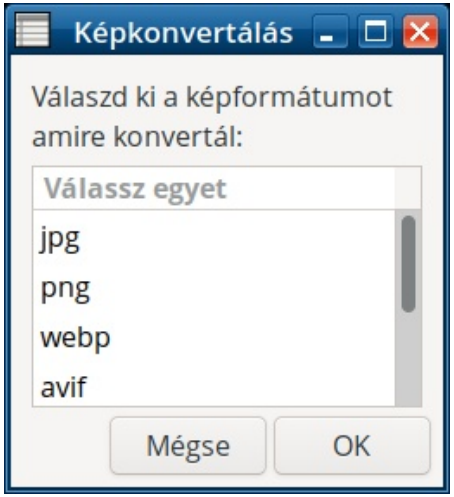
Tehát lent az új, képátméretező szkript látható. A képhez a szövegszerkesztőben hozzáadtam a sorszámozást, hogy könnyebb legyen követni.

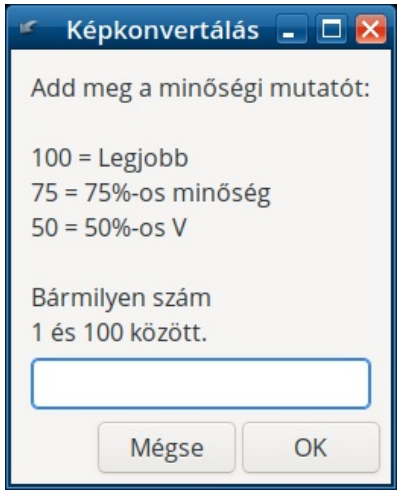
```
1  #! /bin/sh
2
3  n=1
4
5  RES=`zenity --title="Image Width" --width=150 --height=100 --entry --text="Image Resizer -- Please enter
   the \nimage width in pixels (e.g. 600)"`
6     if [ $? == 1 ]; then
7        exit
8     fi
9
10 EXT=$(zenity --list --column="Select One" --title="Convert Image" --width=250 --height=250
   --text="Select the image format\nto convert to:" jpg png webp avif tiff bmp gif)
11    if [ $? == 1 ]; then
12       exit
13    fi
14 declare -l EXT
15 EXT=$EXT
16
17 QUAL=`zenity --title="Convert Image" --width=250 --height=250 --entry --text="Enter the quality
   level:\n\n100 = Full Quality\n75 = 75% Quality\n50 = 50% Quality\n\nAny number between\n1 and 100
   accepted."`
18    if [ $? == 1 ]; then
19       exit
20    fi
21    if [[ ( $(($QUAL)) -lt 1 ) || ( $(($QUAL)) -gt 100 ) ]]; then
22       zenity --title="Convert Image" --width=200 --height 100 --error --text="Exiting.\n\nNumber
         entry\nout of range."
23       exit
24    fi
25
26 DIR=`zenity --title="Destination Directory" --entry --text="Enter the destination directory.\n\nEnter ./
   for the current directory.\nEnter ../ for the parent directory.\n"`
27    if [ $? == 1 ]; then
28       exit
29    fi
30    if [ ! -d $DIR ]; then
31       mkdir $DIR
32    fi
33
34 if [ $EXT == "png" ];
35    then
36       BACKGROUND="none"
37       QUALITY=$(($QUAL / 10))
38    else
39       BACKGROUND="white"
40       QUALITY="$QUAL"
41 fi
42
43 REZ=$((RES-2))
44
45 for file in $@; do
46    if [ ! -e $file ]; then
47       continue
48    fi
49
50    name=$( echo $file | cut -f1 -d.)
51    convert $file -units pixelsperinch -resample 72x72 -resize $REZ -quality $QUALITY -fuzz 20%
      -background $BACKGROUND -bordercolor black -border 1x1 $DIR/${name}_$RES-$QUALITY.$EXT
52    echo $(($n * 100 / $#))
53    echo "# Processing file: $file"
54    let "n = n+1"
55
56 done | (zenity  --progress --title "Resizing..." --percentage=0 --auto-close --auto-kill)
57
58 exit 0
```

A szkript természetesen a tipikus bash szkriptindítóval kezdődik, majd a 3. sor beállít egy változót „1” értékre. Az **5.-től 8. sorig** állítod be az átméretezett kép szélességét pixelben. Kisebb szélesség, kisebb fájlméretet eredményez és

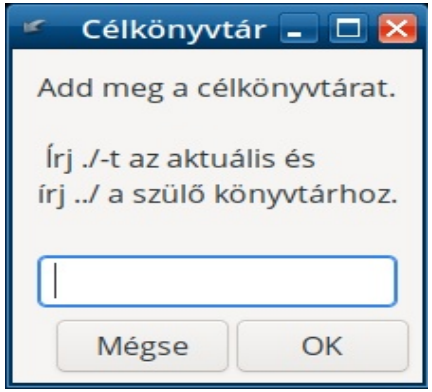


viszont, nagyobb szélesség, nagyobb fájlméretet produkál. Azt tanácsolom, hogy NE haladd meg túlságosan az eredeti fájl méretét, vagy valószínűleg a kép pixelezését fogod tapasztalni. Vannak más eszközök, amik sokkal jobban képesek megnagyobbítani a képet … de ez már egy másik cikkre tartozik.





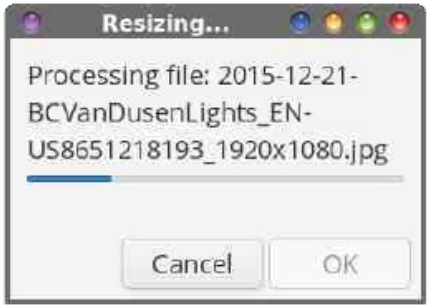
A **10.-től 15.-ig sor** teszi lehetővé a fájlformátum kiválasztását, amire a képe(ke)t konvertálod. A **17.-től 24. sorig** lehet meghatározni a képminőség beállításait. Minél nagyobb a szám, a minőség annál jobb és fájlméretet is nagyobb.



A **26-tól 32. sorig** rész teszi lehetővé a tároló könyvtár kiválasztását az átméretezett fájloknak. Ha az eredeti fájlokkal azonos könyvtárban tárolnád, könyvtárként írj „./”-t. Hasonló módon, célkönyvtárnak „../”-t írva az átméretezett képet az eredeti képedet tároló könyvtár szülő könyvtárába fogja elhelyezni, bárhol legyen is az.

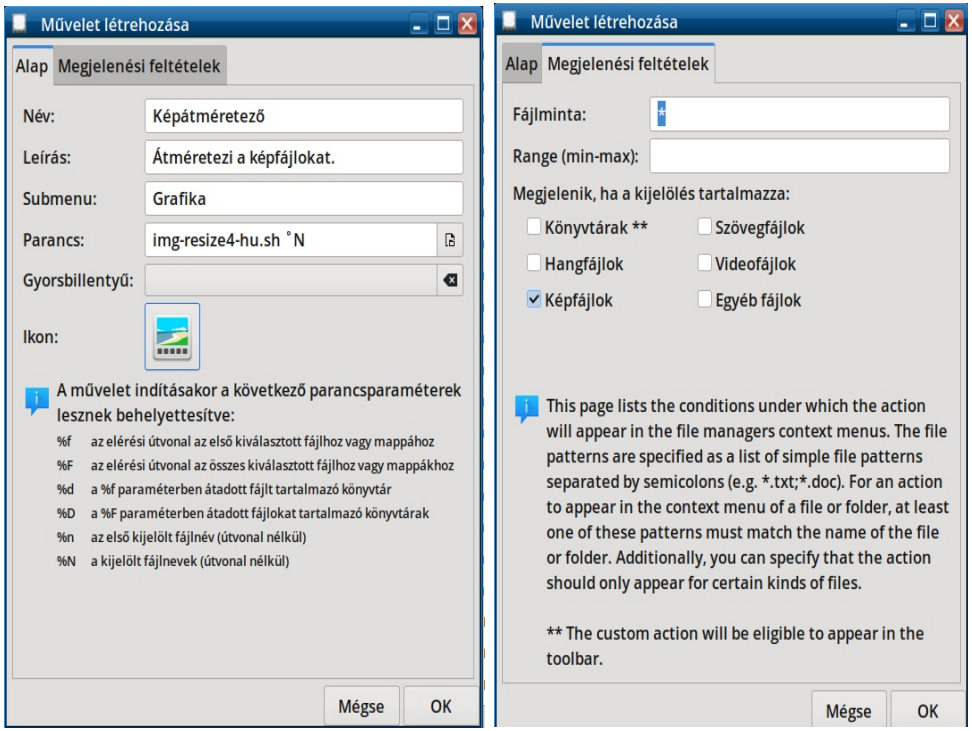
A **34.-től 41.-ig sorok**, attól függően módosítják egy kicsit a kép háttérértékét, hogy .PNG-re, vagy .JPG-re konvertálsz. A 43. sor módosítja a képed felbontását (szélességét), hogy helyet csináljon a szkript által a kép szélei köré húzott egypixeles fekete körvonal elhelyezéséhez.

A **45.-től 54-ig sorok**, ahol a „varázslat” történik. A szkriptnek parancssori paraméterként elküldött fájllista egy „for” ciklusba kerül feldolgozásra, ahol mindegyik fájl új nevet kap, új felbontással, minőségi szinttel és új fájlkiterjesztéssel. A szkriptben a munka dandárját az ImageMagick „**convert”** parancsa végzi el. A legelején, a 3. sorban beállított változó léptetése teszi lehetővé a folyamatjelző megfelelő frissítését.



Az **56. sor** felelős a folyamatjelző és párbeszédablak megjelenítéséért, miközben a szkript a „betáplált” képeket dolgozza fel. Kis számú képen dolgozva lehetséges, hogy észre sem fogod venni a párbeszédablak megjelenítését. Ugyanakkor, ha nagyszámú fájllal dolgozol, örülni fogsz a folyamatjelzőnek. Az **58. sor** léptet ki a szkriptből tisztán, amint az összes fájl feldogozásra került.

Ahogy korábban említettem, a szkriptet úgy terveztem, hogy működjön akár parancssorból, akár Thunar egyéni műveleteként. A Thunar egyéni művelet beállításához a műveleti ablak valahogy a következő képen látható módon nézhet ki.



A szkript ezen verziójában minden úgy működik, ahogy azt terveztem. A szkript kicsi és G Y O R S.

Az „önálló” szkript

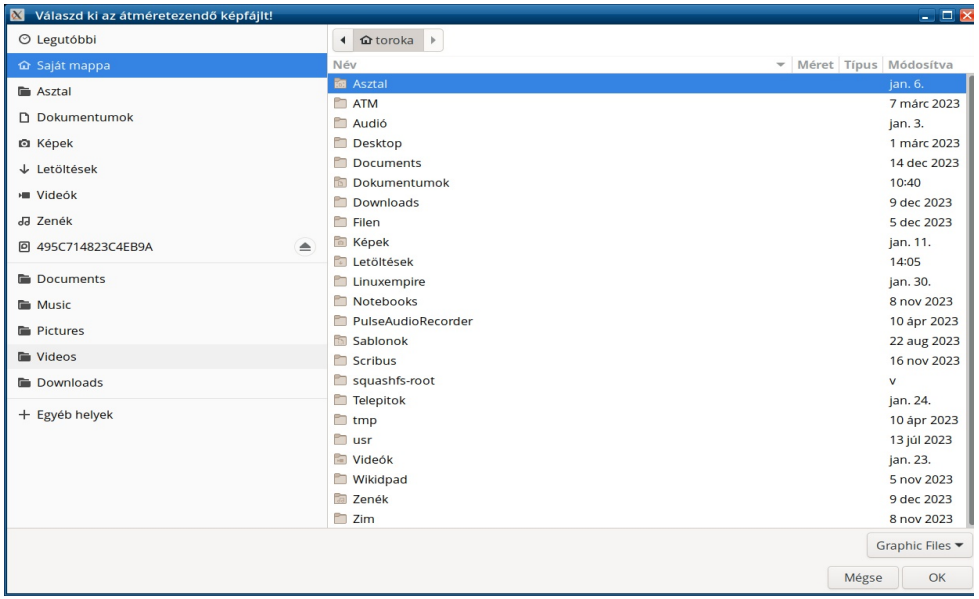
A képátméretező szkriptem frissítésekor elhatároztam, hogy megpróbálok olyan önálló szkriptet készíteni, ami futtatható a panelen, vagy az asztalon lévő indítóból is. Látszólag a korábbi verzió ***MINDEN*** működési funkcióját megtartotta, de van némi különbség, amit a szkript átnézése közben fel fogok fedni. Végül is, az előző verzióban minden szükséges adatot a parancssoron keresztül kellett bevinni. Ebben az önálló verzióban ezeket a szükséges adatokat teljes egészében grafikus felületen (GUI) kell „betáplálni”.

Ahogy az előbbi szkriptet, ezt is le lehet tölteni a Magazin honlapjáról. És, ahogy azt az előző szkriptnél kellett, itt is töröld a „.txt” fájlkiterjesztést, mentsd oda, ahol általában a szkriptjeidet menteni szoktad és tedd végrehajthatóvá. A szkript ezen verziójának **Img-resizer5b** a neve.

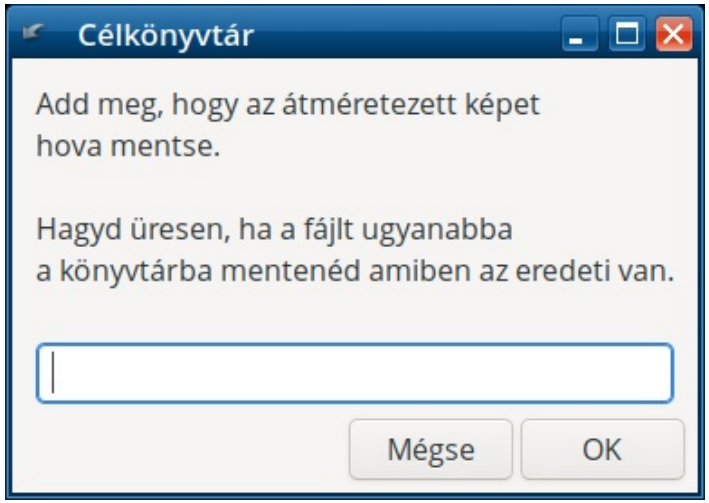
A szkript nagy része változatlan, megegyezik az előző verzióval pár észrevehető különbséggel. Először, íme az önálló programként működő szkript.

```
#! /bin/sh

n=1

FILES=$(zenity --file-selection --multiple --separator=" " --filename "${HOME}/" --file-filter='Graphic Files | *.png *.PNG *.jpg *.JPG *.w
   if [ $? == 1 ]; then
      exit
   fi
read -r -a "FILES" <<< $FILES
count=${#FILES[@]}

RES=`zenity --title="Image Width" --width=150 --height=100 --entry --text="Image Resizer -- Please enter the \nimage width in pixels (e.g. (
   if [ $? == 1 ]; then
      exit
   fi

EXT=$(zenity --list --column="Select One" --title="Convert Image" --width=250 --height=250 --text="Select the image format\nto convert to:"
   if [ $? == 1 ]; then
      exit
   fi
declare -l EXT
EXT=$EXT

QUAL=`zenity --title="Convert Image" --width=250 --height=250 --entry --text="Enter the quality level:\n\n100 = Full Quality\n75 = 75% Qual:
   if [ $? == 1 ]; then
      exit
   fi
   if [[ ( $(($QUAL)) -lt 1 ) || ( $(($QUAL)) -gt 100 ) ]]; then
      zenity --title="Convert Image" --width=200 --height 100 --error --text="Exiting.\n\nNumber entry\nout of range."
      exit
   fi

DIR=`zenity --title="Destination Directory" --width=350 --entry --text="Enter the directory to save your\nresized images in.\n\nLeave the e
   if [ $? == 1 ]; then
      exit
   fi
CURR_PATH=$(dirname "${FILES[0]}")
NEW_DIR=$CURR_PATH"/"$DIR
   if [ ! -d $NEW_DIR ]; then
      mkdir $NEW_DIR
   fi

if [ $EXT == "png" ];
   then
      BACKGROUND="none"
      QUALITY=$(($QUAL / 10))
   else
      BACKGROUND="white"
      QUALITY="$QUAL"
fi

REZ=$((RES-2))

for file in "${FILES[@]}"; do
    if [ ! -e $file ]; then
        continue
    fi

    nopath=$(basename "$file")
    name=$( echo $nopath | cut -f1 -d.)
    convert $file -units pixelsperinch -resample 72x72 -resize $REZ -quality $QUALITY -fuzz 20% -background $BACKGROUND -bordercolor black
    echo $(($n * 100 / $count))
    echo "# Processing file: $file"
    let "n = n+1"

done | (zenity  --progress --title "Resizing..." --percentage=0 --auto-close --auto-kill)

exit 0
```

Tehát, ahelyett hogy újra átvennénk azt, amit az előző szkripttel kapcsolatban mondtam, csak a különbségekről fogok beszélni.



A Zenity fájlválasztó párbeszéde az **5.-től 9.-ig sorok** között jelenik meg. A Zenity-parancs CSAK azoknak a fájloknak (és könyvtáraknak) a megjelenítésére korlátozódik, amik kiterjesztése: PNG, JPG, WEBP, AVIF, GIF, BMP és TIFF. Azt is megcsináltam, hogy működjön olyan képekkel, amik fájlkiterjesztésében csupa kisbetű, illetve olyan képekkel, amik kiterjesztésében csupa nagybetű szerepel. Azt is lehetővé teszi, hogy több fájlt válasszunk ki átméretezésre. A fájlnevek közötti elválasztást a szóköz jelenti. Ha nem módosítottamvolna, az elválasztó „|” lenne alapbeállításban. Ugyanakkor a read (olvasás) parancs szóközt vár elválasztóként, így a szóközre váltás lehetővé teszi a read paranccsal tömb létrehozását. A 9. sorban a kiválasztott fájlokból tömb készül a további feldolgozáshoz.



A **33.-tól 41. sorokig** meg kellett változtatni pár dolgot. Először módosítani kellett a „Destination Directory” (célkönyvtár) párbeszédet. Másodszor meg kellett szerkeszteni az új képhez a megfelelő útvonalat és létrehozni a célkönyvtárat, ha nem lenne. A szkript önálló változatában nem adhatod meg (../) formában célkönyvtárként a szülő könyvtárat. Ugyanakkor, ha ez eredeti képfájlokkal azonos könyvtárba szeretnéd menteni az átméretezett képeket (emlékezz, az eredeti fájlok a felülírás ellen védettek), egyszerűen hagyd üresen a beviteli mezőt.

Amikor a Zenity fájlválasztó párbeszédablakában kiválasztottuk a fájlokat, a visszaadott fájlnevek teljes útvonalleírást kapnak. Tehát, mindahhoz hogy megfelelően létrehozzuk az új könyvtárat és megfelelő útvonalat készítsünk a célfájlhoz, szét kell bontanunk és újra felépítenünk a megfelelő útvonalat a célfájl számára. A CURR_PATH változó veszi az tömb első kiválasztott fájlját (${FILES[0]}) a „dirname” paranccsal. Emlékezz, a kiválasztott fájlokkal kapcsolatos összes információt tömbbe szerveztük. Vagyis csak annyit teszünk, hogy annak az első fájlnak az útvonal-információit bemásoljuk a CURR_PATH változóba. Ezután létrehozzuk a megfelelő útvonalat a célfájl számára úgy, hogy megfelelő módon összerakjuk a $CURR_PATH-t és az $DIR-t (amit a Célkönyvtár párbeszédben határoztunk meg) és mentjük a $NEW_DIR nevű változóba. Ezt a változót később fogjuk használni, amikor az ImageMagick „convert” parancsa létrehozza az új kimeneti fájlt.

Az **54. sorra** előreugorva, a „for” ciklusunkat át kell alakítanunk úgy, hogy a 9. sorban létrehozott tömbből vegye a fájlnevet. A megfelelő fájlnévre irányuló „szerkesztési” erőfeszítéseink részeként, a feldolgozás során a FILES tömb minden egyes fájljának útvonal-információját lecsupaszítjuk (alapnév). Ezután a $NEW_PATH-t egyesítjük az éppen feldolgozott fájlnévvel és a többi fájlnévleíró adattal, hogy elkerüljük az eredeti fájl felülírását, így létrehozva a teljesen érvényes és megfelelő útvonalat az ImageMagick „converter” parancsa részére az új kimeneti fájlhoz.

A szkript majdnem mindegyik fennmaradó sora teljesen megegyezik a korábbi verzióval. A szkript jól teljesít a képek átméretezésében. Próbáltam, próbáltam és próbáltam kiakasztani, nem jártam sikerrel a próbálkozásokkal. Minden alkalommal és helyzetben, amikor próbáltam, jól működött.

Ha nem akarod, még csak célkönyvtárat sem kell megadnod. Csak eszembe jutott, hátha néhány felhasználó szeretné ezt a kellemes „tulajdonságot”. Az átméretezett képek ekkor az eredeti fájlokat tartalmazó könyvtárba kerülnek. Mivel az eredeti fájlok védve vannak a felülírással szemben, ez nem olyan nagy gond. Később, ha meggondolod magad, egyszerűen csak be kell lépned abba könyvtárba, ami az eredeti és az átméretezett fájljaidat tartalmazza és kézzel átvinni az átméretezett fájlokat másik könyvtárba, ha kívánod. De az a képesség, hogy új könyvtárat készíthetsz az átméretezett fájloknak, megszabadíthat attól, hogy kézzel kelljen elvinni az átméretezett fájlokat később, így máskor időt takarít meg. Emellett segít a dolgokat tisztán és rendben tartani.

Hátrányok

Teljesen őszintén, legjobb tudomásom szerint, csak EGY dolgot tudok felhozni ezekkel a szkriptekkel szemben. Ez pedig, hogy a grafikus fájl neve NEM tartalmazhat szóközt. Távolítsd el, vagy cseréld le a szóközt a fájlnevekben, amiket a szkripteknek feldolgozásra átadsz, különben megakadályozhatják a szkript lefutását.

Természetesen „tudod, ez nem Windows”, ezért semmiképpen sem szabad szóközt használni a fájlneveidben. Ha mégis, egy darabig ellehetsz vele, de alkalomadtán vissza FOG ütni neked a semmiből. Ez azért van, mert sok esetben a szóköz elválasztóként (delimiter) működik a Linuxosok által használt sok programban. Ezért jobb, ha kerülöd a szóközt a fájlneveidben, amennyiben lehetséges.

Különösen valószínűtlen, hogy az önálló szkriptverzió egy tömböt, ami szóközöket tartalmazó fájlneveket tartalmaz, megfelelően dolgozzon fel. A „read” parancs számára, ami a tömböt összeállítja, a szóköz elválasztóként működik, így a tömb fogja a helyes útvonalat, vagy fájlnevet tartalmazni a tömb néhány elemére. Nagy valószínűséggel, nem egy teljesen pontos útvonalra és fájlnévre fog mutatni.

Jó ötlet lehet a szkript elejére berakni egy sort (a 4. sor mögé, mielőtt bármilyen másik párbeszéd megjelenne), ami figyelmezteti a felhasználókat, hogy NE használjanak olyan fájlokat, amik fájlnevében szóköz van. Valahogy így nézhetne ki:

zenity --warning --title=”FIGYELEM!” --timeout=15 –text=”NE futtassa a szkriptet olyan fájlokon, amik a nevükben szóközt tartalmaznak!”

Ez megjeleníti a figyelmeztető ablakot a megadott szöveggel és 15 másodperc után automatikusan eltűnik. Természetesen, bármikor lenyomhatod az „OK” gombot, mielőtt a 15 másodperc letelne, eltüntetve a figyelmeztetést hamarabb mint 15 másodperc.

Ha annyira akarod, berakhatsz egy függvényt, vagy parancsot is a fájlnevekben lévő szóközök eltávolítására, vagy lecserélésére. Ezt rátok hagyom, ha meg akarnátok oldani. Mivel én úgy kerülöm a szóközöket a fájlnevekben, mint az „Ördög a szentelt vizet”, számomra ez nem téma és nem veszek magamra ilyen feladatot. Emellett, azt akarom, hogy a szkript „karcsú” legyen, és a szóközöket tartalmazó fájlnevek átalakításának „kezelése” egyáltalán nem teszi a szkriptet olyan karcsúvá, amilyennek lennie kell, vagy amilyennek én szeretném. A „több”, nem mindig jobb.

Összegzés

Remélem, te is olyan hasznosnak találod a szkriptet, mint én. A Magazin főszerkesztőipozíciómban alig múlik el nap úgy, hogy ne használnám az átméretező szkriptek egyikét. Azaz, RENGETEGET használom ezeket.

Nyugodt lehetsz, tökéletesen teszik a dolgukat. Nincs vesztegetni való időm olyan szkriptekre, amik nem működnek megfelelően, vagy tökéletesen és gyanítom, hogy ez rátok is igaz.

(Ford. megj.:** a magyarra átírt szkriptek hamarosan elérhetőek lesznek!**)