PDF 3B rész

PDF-fájlok manipulálása, parancssori stílus

PCLinuxOS Magazine – 2023. február

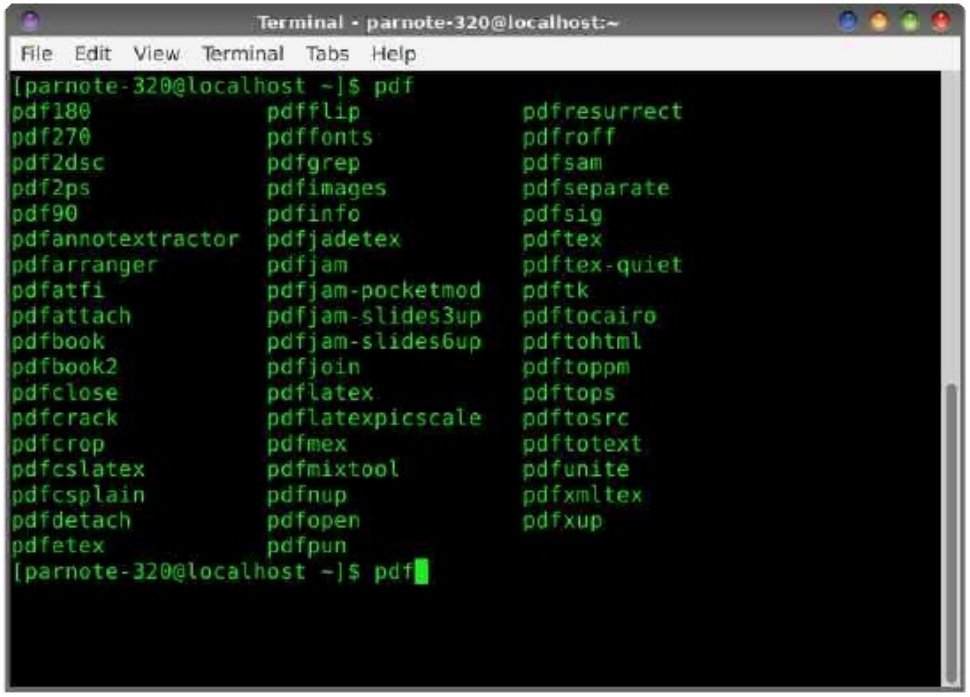
rta: Paul Arnote (parnote)

A PDF-fájlokról a cikksorozatot a 2022. októberi számban indítottam. Abban a cikkben a PDF-fájlok készítéséről beszéltünk. Utána, a 2022. novemberi számban a PDF-fájlok készítéséről volt szó. 2023 januárjában a különféle grafikus PDF-manipuláló eszközök használatáról beszéltünk.

Ez a cikk azt mutatja be, amit a sorozatlezáró cikknek szánok. Most néhány olyan parancssori program használatát nézzük meg, amik a PDF-fájlok manipulálására elérhetőek. Miközben tudom, hogy sokan vannak, akik kerülik a parancssort, mint a mondásbeli bubópestist, ezeket az eszközöket kifejezetten könnyű használni. Emellett ezekkel az eszközökkel lehet a leggyorsabban manipulálni a PDF-fájlokat. A „csinálj egy dolgot, de azt jól” filozófiát követik, amit megannyi Linux-eszköz is követ.

Telepítsük az eszközöket

A PCLinuxOS-rendszered nem tartalmazza alapból az összes ilyen PDF-eszközt. Mivel az utóbbi pár év óta részt veszek a The PCLinuxOS Magazine készítésében, ezek az eszközök a „jó barátaimmá” váltak, amik elfoglalták méltó helyüket az én PDF-es szerszámosládámban. Ugyanakkor, nem kell ahhoz a Magazinnal közvetlenül foglalkoznod, hogy megtaláld ezen eszközök mindennapi hasznát, ahogy azt majd az eszközök leírásához társított példákból is látni fogod.



Sok ilyen PDF-eszköz elérhető a gépeim egyikén.

A fenti kép bemutatja az egyik gépemen elérhető sok parancssori PDF-eszköz közül néhányat. A felhasználóknak nem kell minden egyes ilyen eszközt külön-külön, egyenként telepíteni. Ehelyett, a **pdftk** és a **pdfjam** telepítése a legtöbb ilyen eszközt is hozza magával. A két csomag egyaránt elérhető a PCLinuxOStárolójában és Synaptic-kal telepíthető.

Ugyanakkor, óvatosságra intenélek. A **pdfjam** csomagjának függőségei között a TexLive is szerepel, és a TexLive csomagja elég méretes. Érdemes ellenőrizni a gyökérparticiódat, megbizonyosodni arról, hogy elegendő hely áll-e rendelkezésre? A sosem használt programok leszedése sokat segíthet, ám azzal légy tisztában, az egyes eltávolítandó fájlok mit visznek magukkal. El kell kerülni, hogy az eltávolítandó program eltávolítson olyan, látszólag hozzá nem tartozó programot, vagy csomagot is. Például az „úti” laptopomon (amin ezek az eszközök vannak) meg kellett szabadulnom sosem használt dolgoktól, mint

a Thundetbird levelező profram és a rendszerhez települő Bluetooth eszközök (az adott laptop nem rendelkezik bluetooth-képességgel, vagyis teljesen szükségtelen a jelenlétük azon a számítógépen), különben nem lett volna elegendő helyem a TexLive-csomagok telepítésére.

Ha a PDF-fájlban hivatkozások szerepelnek nem biztos, hogy a manipulált PDF-fájlban is megmaradnak. Érdemes ellenőrizni a hivatkozások meglétét, mielőtt elküldenéd a módosított PDF-fájlt. Kapásból megmondhatom, hogy a **pdfjam**-mel manipulált fájlok a hivatkozásaikat NEM fogják megőrizni. A készítője maga vallotta be mindezt a GitHub-os oldalán.

Nem fogom az összes eszköz mindegyikét kitárgyalni. Helyette inkább azokkal az eszközökkel fogok foglalkozni, amik szerintem az átlag felhasználó számára leginkább fontosak lehetnek. Ezek közül az eszközök közül néhány kifejezetten a TexLive-hoz, vagy a LaTeX-hez kapcsolódik, amik engem vagy nem érdekelnek, vagy nincs rájuk szükségem, illetve nem ismerem az adott kiadványkészítő csomagokat. Korábban már belenéztem azokba és akkoriban úgy találtam, hogy a képességeik az én igényeimet jóval meghaladják. Neked lehet, hogy más kell. Amennyiben így lenne, akkor valószínűleg azt is jobban tudod mint én, hogy azokat a LaTeX-, vagy TexLive-eszközöket hogyan kell használni.

Most akkor a való életből vett példán keresztül nézzünk meg néhányat ezek közül a parancsok közül, amik lehetővé teszik számunkra a PDF-fájlok manipulálását. Közülük néhány furának tűnhet, de a többségük nagyon hasznos. Magától értetődő feladat lehet néhány parancs „automatizálása” bash-szkripttel, sőt Zenity-párbeszéddel kiegészítve. De most az egyszerű parancssori használatukra fogunk fókuszálni.

Vonjunk össze több PDF-et egybe: pdfjoin és pdfunite

Ez olyan eszköz, amit elég rendszeresen használok. Gyerekeim iskolája a menza menüjét online publikálja valami fura nézőben, ami vagy a reggelit és ebédet, vagy a menüket hajlandó mutatni, de egyszerre csak egyet. Sokkal egyszerűbb PDF-fájlt készíteni az egyes menükből és azokat kombinálni egyetlen közös PDF-be. Gyerekeim nem igazán válogatósak, de időnkén összejön valami UTÁLATOS ételválaszték. Azokon a napokon inkább visznek be magukkal ebédet az iskolába ahelyett, hogy az utálatos napi kínálatot ennék. Mint olyanok, segítünk nekik a menü átböngészésében, hogy biztosan olyasmit egyenek, amit szeretnek. Ellenkező esetben éhen maradnak, aminek az oktatásuk látja kárát. Végül is nehéz üres, korgó vagy éhes gyomorral tanulni.

Ehhez megnyitom a fura online nézőt és kiválasztom a nyomtatás menüt. A kimenetnek a „Mentés PDF-be”-t választom a Firefox nyomtatás-választó dobozában. Ezután, minden menünek saját nevet adok. Például, a januári éttermi menü letöltésekor úgy neveztem el azokat, hogy Jan2023Breakfast.pdf, Jan2023Lunch.pdf és Jan2023Snack.pdf

Ezután belépek abba a könyvtárba, ahová elraktam azokat és beírom a **pdfjoin** parancsot ily módon (minden egy sorba kerül):

```
pdfjoin Jan2023Breakfast.pdf Jan2023Lunch.pdf
Jan2023Snack.pdf -o Jan2023-Menu.pdf
```

Íme a művelet teljes kimenete:

```
[parnote-320@localhost ~]$ cd Documents
[parnote-320@localhost Documents]$ pdfjoin
Jan2023Breakfast.pdf Jan2023Lunch.pdf Jan2023Snack.pdf -o
Jan2023-Menu.pdf
                              ----
  pdfjam: This is pdfjam version 3.03.
  pdfjam: Reading any site-wide or user-specific defaults...
        ##
        ## From /etc/pdfjam.conf:
        ##
        tempfileDir='/var/tmp'
        tidy='true'
        keepinfo='false'
        checkfiles='false'
        paper='a4paper'
        outFile="$pwd"
        suffix='pdfjam'
        landscape='false'
        twoside='false'
        preamble=''
  pdfjam: Effective call for this run of pdfjam:
      /usr/bin/pdfjam --fitpaper 'true' --
rotateoversize 'true' --suffix joined --outfile Jan2023-
```

```
Menu.pdf -- Jan2023Breakfast.pdf - Jan2023Lunch.pdf -
Jan2023Snack.pdf -
      pdfjam: Calling /usr/bin/pdflatex...
pdfjam: Finished. Output was written to 'Jan2023-
Menu.pdf'.
```

A PDF-fájlokat olyan sorrendben sorold fel, amilyenben az új, egyesített PDF-fájlban látni akarod. Az PDF-fájl meghatározása a **-o** parancssori kapcsolóval törénik, amit az új PDF-fájl neve követ paraméterként. A fenti példámban azt Jan2023-Menu.pdf-nek neveztem el. Ezután az új PDF-fájlt az asztalomra rakom, ahonnan gyorsan megnyithatom, amikor a gyerekek az iskolába készülnek. És minden ott van, egyetlen, könnyen olvasható PDF-fájlban.

Most beírhatnám az egyes fájlok nevét a teljes elérési útvonalukkal is ahelyett, hogy cd paranccsal belépnék abba a könyvtárába, ahova a fájlokat mentetem. De én inkább a cd paranccsal belépést a fájlok mentési könyvtárába preferálom, mivel minden egyes fájlnak megadni a teljes elérési útvonalát nehezen kezelhetővé teszi a parancsot és túl sok hibalehetőséggel jár.

A **pdfunite** parancssori eszköz a Poppler-fejlesztőktől származik és elég hasonlóan működik, mint a pdfjoin. Van azonban pár apró különbség. Például incs szükség a "-o" paraméter használatára a kimeneti fájl meghatározásához. Ehelyett csak felsorolod az „egyesítendő” (unite) PDF-fájlokat és a kimeneti PDF-fájl nevét legutolsó paraméterként írod be. A pdfunite verziószámát megjelenítő „-v” és a súgó tartalmát megjelenítő „--help” opciókon kívül nincs más parancssori kapcsolója az eszköznek.

A pdfunite kezeléséhez teljesen ugyan olyan módon közelítek, mint ahogy a pdfjoin-hoz is (amit a pdfjam szerzője készített), azaz a cd paranccsal belépek az egyetlen PDF-fájlba egyesítendő fájlokat tartalmazó könyvtárba és a parancsot onnan futtatom. Akár a pdfjoin-nál is lehetett, a fájlneveket teljes értékű útvonalakkal ,egadva is használhatod, de sokkal egyszerűbb mindent egyetlen könyvtárban tartani és azokat a könyvtáron belül összevonni .

Adjunk egy kis csavart a dolgokhoz: PDF-oldalak forgatása

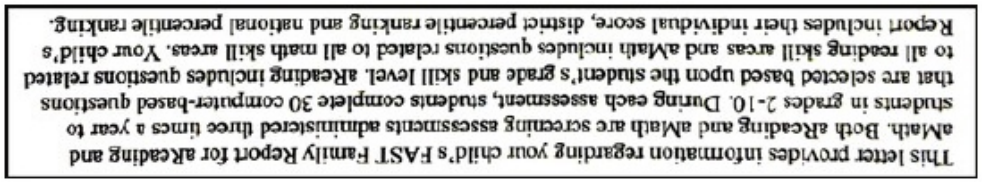
Nézzünk meg egy másik helyzetet. A fiam összes IEP-jéből (Egyéni Oktatási Terv), a negyedévi IEP-frissítésről, teljesítési jelentésről (besorolási kártyák), a kerületi szintű felmérésről ... másolatot tartunk. Valójában mindenről van digitális másolatunk, ami segít követni az iskolai előrehaladást. Igen, ugyanez igaz a lányomra is. Neki nincs IEP-je, de minden egyéb dokumentumáról feljegyzésünk van, amivel az ő fejlődését is követni tudjuk.

A helyi iskolakörzetet is tájékoztattuk az otthoni egyénenkénti nyilvántartásunkról és megígérték, hogy ezeket a dokumentumokat továbbítják nekünk e-mailben, PDF-formátumban. Noha belegyeztek az előbbiekbe, nem mindig teszik meg. Így, amikor a gyerekek hazahozzák a dokumentumok papíros változatát, két lehetőségem van. Első, hogy felveszem a kapcsolatot az aktuális tanárral és elküldetem a dokumentum digitális változatát. A másik lehetőség, hogy én digitalizálom.

Noha **van** síkágyas szkennerem, sokkal egyszerűbb a hazahozott papír alapú dokumentumot az Adobe Scan-nel a mobiltelefonommal beszkennelni (Androidra a Google Play áruházban, iOS-eszközre az iOS alkalmazásboltban, vagy az Adobe-termékek oldalán az Adobe Scan-nél itt érhető el). Gyorsabb és könnyebb, mint ugyanezt egy számítógéphez kötött szkennerrel elvégezni, előbb bekapcsolva a gépet, stb. stb. stb. Ezután a kapott PDF-fájlt elküldöm e-mailben a telefonomról a személyes postaládámba. Az Adobe céggel szembeni minden utálatom ellenére, az Adobe Scan alkalmazás kifejezetten jól működik, létrehozva a nekem kellő PDF-fájlokat, sokkal könnyebben, mintha síkágyas szkennert használnék. És ingyenes.

UGYANAKKOR – és itt van az, amikor a következő parancssori PDF-eszköz színpadra léphet – néha a PDF-fájl rosszul forgatva jelenik meg. megtehetném, hogy egyszerűen

újraszkennelem az iratot, de az elég frusztráló lenne. Az újraszkennelés szintén tovább tart, mint egyszerűen parancssori eszközökkel „kijavítani' a rossz forgatást, laptájolást.


This letter provides information regarding your child's FAST Family Report for aReading and aMath. Both aReading and aMath are screening assessments administered three times a year to students in grades 2-10. During each assessment, students complete 30 computer-based questions that are selected based upon the student's grade and skill level. aReading includes questions related to all reading skill areas and aMath includes questions related to all math skill areas. Your child's Report includes their individual score, district percentile ranking and national percentile ranking.

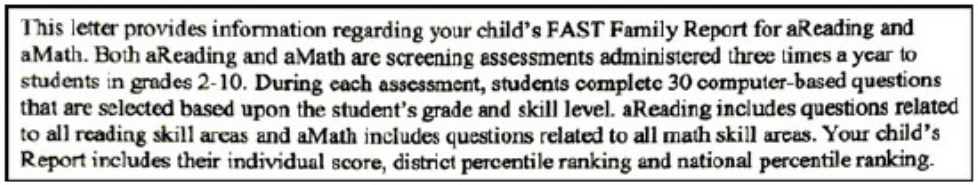
Az eredeti szkennelés rossz tájolással

Miközben egyszerűen kinyomtathatnám a nyomtatómon és úgy elforgathatnám a kinyomtatott papírt, mert a PDF-fájl a jelenlegi állapotában nehezen olvasható. Tehát, el kell forgatni a szöveget. A **pdfjam**-hez három olyan eszköz tartozik, amivel a szöveg elforgatható. Ezek a **pdf90**, **pdf180** és a **pdf270**. Az egyes parancsok végén a szám mutatja a szövegforgatás szögértékét, órajárásával ellentétes irányban haladva. Ha a forgatásra úgy nézel, mint ahogy a órák számai elhelyezkednek a számlapon, akkor a pdf90 a kilencórai állást, a pdf180 a határai állást és a pdf270 a háromórai helyzetet jelenti.

Mivel a dokumentum a kívánt helyzetéhez képest 180 fokkal elforgatva jelenik meg, a pdf180 eszközt használjuk a dokumentum „javítására". A következőkben látható a parancs és a pdf180 kimenete:

```
[parnote-320@localhost 2022-2023-Third-Grade]$ pdf180 Ryan-
FAST-Scores-3rd-Grade-01-06-23.pdf
----
  pdfjam: This is pdfjam version 3.03.
  pdfjam: Reading any site-wide or user-specific defaults...
                  ##
                  ## From /etc/pdfjam.conf:
                  ##
                  tempfileDir='/var/tmp'
                  tidy='true'
                  keepinfo='false'
                  checkfiles='false'
                  paper='a4paper'
                  outFile="$pwd"
                  suffix='pdfjam'
                  landscape='false'
                  twoside='false'
                  preamble=''
                  pdfjam: Effective call for this run of pdfjam:
                  /usr/bin/pdfjam --suffix rotated180 --angle
                  '180' --fitpaper 'true' -- Ryan-FAST-Scores-3rd-Grade-01-06-
                  23.pdf -
                  pdfjam: Calling /usr/bin/pdflatex...
                  pdfjam: Finished. Output was written to '/home/parnote-
                  320/Documents/Ryan/School/2022-2023-Third-Grade/Ryan-FAST-
                  Scores-3rd-Grade-01-06-23-rotated180.pdf'.
```

A PDF-fájl forgatását végrehajtó aktuális parancsot (egyetlen sorba kell beírni) a terminál szövegéből pirossal emeltem ki. Vedd észre, hogy nincs szükség kimeneti fájl megadására. Ennek az az oka, hogy az eredeti fájlnevet őrzi meg, a „-rotated180"-at (180-nal elforgatva) biggyesztve a végére. Ezzel úgy forgathatod el, hogy nem kell tartanod az eredeti fajl elvesztésétől.


This letter provides information regarding your child's FAST Family Report for aReading and aMath. Both aReading and aMath are screening assessments administered three times a year to students in grades 2-10. During each assessment, students complete 30 computer-based questions that are selected based upon the student's grade and skill level. aReading includes questions related to all reading skill areas and aMath includes questions related to all math skill areas. Your child's Report includes their individual score, district percentile ranking and national percentile ranking.

Tádá! Most a szöveg a PDF-fájlban az olvasáshoz megfelelően áll. Az átalakítás gyors (gyorsabb mint újraszkennelni) és könnyű (könnyebb, mint bekapcsolni a gépet, amihez a szkenner kapcsolva van).

Kapjunk több információt, tudjunk meg többet: pdfinfo

Hiszed, vagy sem, a **pdfinfo** nem része sem a pdfjam-nek, sem a pdftk-nak. Sokkal inkább a poppler PDF-könyvtárfájlok részének tűnik. Ám ez nem csökkenti az értékét.

Anélkül, hogy a legtöbb felhasználó tudná, a PDF-fájlokban sokkal több információ rejtőzik, mint ami egy PDF-nézőben látható. És ez az eszköz igazából nem manipulálja a PDF-fájlt, mivel arra készült, hogy feltárjon pár, a PDF-fájlba beágyazott olyan információt, amit valójában sosem láthatsz. A feltárt információk csupán tájékoztató természetűek.

```
[parnote-320@localhost FinalAssembly]$ pdfinfo 2022-12.pdf
Title:
Subject:
Keywords:
Author:
Creator:         Scribus 1.4.8
Producer:        Scribus PDF Library 1.4.8
CreationDate:    Mon Dec  5 04:30:10 2022 CST
ModDate:         Mon Dec  5 04:30:10 2022 CST
Custom Metadata: no
Metadata Stream: no
Tagged:          no
UserProperties:  no
Suspects:        no
Form:            AcroForm
JavaScript:      no
Pages:           37
Encrypted:       no
Page size:       792 x 612 pts (letter)
Page rot:        0
File size:       10758380 bytes
Optimized:       no
PDF version:     1.4
```

Tegyük fel, hogy kapsz egy PDF-et … valahonnan, vagy valakitől … és szeretnéd tudni, milyen programot használt a PDF készítője a PDF-fájl elkészítéséhez. Egyszerűen futtasd a **`pdfinfo <filenév>`** parancsot és meg kell jelenjen valami hasonló kimenet, mint ami fent látható. Az információban benne van, hogy PDF-fájl készítésére milyen programot használtak, a lapmérete, a PDF mérete byte-okban megadva, készítésének és módosításának dátuma és az alkalmazott PDF verziója. A példáról készült képernyőképen a felsorolt egyéb információk mellett az is látható, hogy a The PCLinuxOS Magazine 2022. decemberi száma Scribus-szal készült

A továbbiakban néhány egyéb példa látható a képernyőkép-részleteken.

```
[parnote-320@localhost PDF]$ pdfinfo Dec2022-Korte-Menu.pdf
Creator:         TeX
Producer:        pdfTeX-1.40.24
CreationDate:    Wed Nov 30 00:31:34 2022 CST
ModDate:         Wed Nov 30 00:31:34 2022 CST
```

Ez a PDF-fájl pdfTex-szel készült, pdfjoin parancsot alkalmazva.

```
[parnote-320@localhost PDF]$ pdfinfo The_Art_of_Resawing___Popular_Woodworking.pdf
Producer:        GPL Ghostscript 9.22
CreationDate:    Sun Aug  7 10:20:10 2022 CDT
ModDate:         Sun Aug  7 10:20:10 2022 CDT
```

Ezt a PDF-fájlt a GPL Ghostscript készítette, a CUPS-PDF nyomtatómeghajtót használva.

```
[parnote-320@localhost Documents]$ pdfinfo john-receipt-lake-well.pdf
Creator:         Writer
Producer:        LibreOffice 6.0
CreationDate:    Mon Mar 29 10:52:28 2021 CDT
```

Ez a PDF-fájl LibreOffice Writer-rel készült.

```
[parnote-320@localhost 2022-2023-Third-Grade]$ pdfinfo Ryan-FAST-Scores-3rd-Grade-0
1-06-23.pdf
Creator:         Adobe Scan for Android 22.12.16-regular
Producer:        Adobe Scan for Android 22.12.16-regular
```

Ezt a PDF-fájlaz okos telefonomon az Adobe Scan készítette.

A verziószámokat megnézve (itt nem látható), a legtöbb PDF-fájl vagy az 1.4-es, vagy az 1.5-ös verzióval készül. Egy kicsit meglepő volt látni, hogy az Adobe Scan a régebbi, 1.3-as verziót használja. Nem igazán tudom, miért nem egy frissebb verziót használnak. Ezt egyszerűen beírom az érthetetlen „fejcsóválós dolgok közé”. Biztos van valami oka, amiről nem tudok, az Adobe pedig nem mondja.

Rakjuk fel az Internetre: pdftohtml

A **pdftohtml** egy másik eszköz, ami a Poppler-fejlesztőktől származik. Ahogy az a nevéből is gondolható, PDF-fájlokat konvertál webböngészőben megjelenítésre alkalmas alap html-fájlokká. A html-fájl(oka)t minimálisan megszerkesztve elérhető, hogy az(ok) az eredeti PDF-re nagyon hasonló megjelenésű(ek) legyen(ek). Én magam nem töltenék túl sok időt azzal, hogy úgy alakítsam át, maitől majd pontosan úgy nézzen ki, mint a PDF. Szerény véleményem szerint, olyan harcba bonyolódnánk bele, amit nem nyerhetünek meg egyetlen alapvető okból: két egymástól eltérő médiumról van szó. Ha csak nem vagy egy önkínzó típus, nem valószínű, hogy teljesen egyforma kinézetűvé tudod alakítani azokat. Egyszer-egyszer sikerülhet, de az esetek többségében nem éred el a célodat.

A maga legegyszerű formájában a parancs kifejezetten könnyen használható. Csak a pdftohtml parancs után be kell írni a HTML-é átalakítandó PDF-fájl nevét, pl.

pdftohtml <filenév.pdf>. Vannak további parancssori opciók is , de azok felfedezését rád hagyom úgy, hogy beírod a **pdftohtml --help**-et a parancssorba. Ugyanakkor, a parancs a maga egyszerű formájában is elvégzi a feladatát.

Amikor a parancsot futtatod, három HTML-fájl készül. Egy a HTML index-keretének, egy a HTML tartalmi keretének és egy pedig az, ami összekapcsolja ezeket. Az utóbbi azt a nevet kapja, ami PDF-fájlé is volt, mivel azt kijelölve nyitod meg a HTML-fájlt a webbögésződben. Az index-keret neve ugyanaz lesz, mint a PDF-é, de egy „_ind” függeléket kap az alap fájlnév mögé. A tartalmi keret neve is azonos lesz a PDF-fájléval, de egy „s” kerül az alap fájlnév mögé.

A viszonylag egyszerű PDF-dokumentumok esetében a pdftohtml csodás munkát végez a HTML-fájl készítésében. Még a képek PDF-ből HTML-be importálásában is meggyőző (de nem tökéletes) munkát végez. Megvannak a korlátai. A legjelentősebb, amit találtam, hogy tapasztalatom szerint NEM kezeli jól a táblázatok adatait. A táblázatban a szöveget felismeri, de a kimenete nem lesz táblázat formájú. Sokkal inkább szöveget a HTML-be „ömleszti” és elég nehéz kisilabizálni, melyik adat tartozik melyik másikhoz tartozik.

Vonjuk ki a szöveget a PDF-ből: pdftotxt

A **pdftotxt** hasonlít a pdftohtml-hez, kivonja a szöveges információt a PDF-ből sima ASCII szövegfájlba. Opcióként megadható a készülő szövegfájl neve, de ha ezt nem teszed, akkor a szövegáfjlnak ugyanazt a nevet adja, mint ami a PDF-é volt.

A csupasz formájában a parancs így néz ki:

```
pdftotext <PDF-fájlnév.pdf> <opcionális-text-fájl-fájlneve>
```

Tehát, a következő példában a 2022. novemberi számból „ A főszerkesztő asztaláról …” cikkemet használva a parancsot így adtam ki:

```
pdftotext Nov2022-Welcome.pdf
```

A PDF szövege egy sima ASCII szövegfájlba került és a bemeneti PDF-fel megegyező nevet kapta.

Vannak parancssori opciói (mint az első és az utolsó oldal szövegének kinyerése), de nem szükségesek. Ha nem adjuk meg, hogy az első és az utolsó oldal szövegét szedje ki, akkor a PDF teljes szövegét kivonja sima ASCII szövegfájlba. Rád hagyom a különféle parancssori opciók felfedezését. A kilistázásukhoz írd be **pdftotext --help** a parancssorba.

Szedjük ki azokat a képeket: pdfimages

Lehetnek estek, amikor szükséged lenne egy képfájlra, de az egyetlen példánya, amihez hozzáférsz egy PDF-fájlban van. Készíthetsz képernyőképet a PDF-fájlban megjelenő képről, de van más módszer is.

A Poppler fejlesztőinek jóvoltából a **pdfimages** eszköz kiszedi a képeket a PDF-fájlból. Noha elég szőrszálhasogató a parancssori paramétereit illetően (különösen az útvonal-kifejezéseket tekintve), ha ezen egyszer túllépsz, elég egyszerű a képeket kiszedni.

Az előző példához hasonlóan a 2022. novemberi számból „ A főszerkesztő asztaláról …” cikkemet használtam. Készítettem egy külön alkönyvtárat TestImages néven a képek kiszedéséhez. Ezután a cd paranccsal a PDF-et tartalmazó könyvtárat tettem meg munkakönyvtárnak. Végül kiadtam a következő parancsot:

```
pdfimages -png Nov2022-Welcome.pdf ./TestImages/Image
```

A -png paraméter közli a pdfimages-szel, hogy PNG-fájlokba mentse a képeket. Ha JPG-ben szeretnéd a képeket menteni, akkor a -png helyett „-j”-t írj. A képek menthetők TIFF-ként is, csak cseréld le a „-png”-t „-tiff”-re. Ha kihagyod a -j, -png, vagy -tiff opciót, akkor monokróm képeket PBM-ként menti, a színes képeket pedig PPM-fájlokként.

Ezekkel nincs semmi baj, noha nem a legelterjedtebb grafikus formátumok a világban. Én vagy PNG-, vagy JPG-fájlokkal szeretek dolgozni.

Hasonlóképpen, ha a „-all” parancssori kapcsolót használod, akkor a PDF képeit az összes általa menthető képformátumban kimenti. Ha szeretnéd kilistázni az elérhető képformátumokat, használd a -list paramétert, miközben a képekhez tartozó fájlneveket letilthatod pl. **pdfimages -list <pdf-fájlnév>.**

Ez után megadjuk a PDF-fájlt, amiből kiszednénk a képeket. Végül megadjuk az útvonalat és a képeknek szánt nevet. Az előző példában a képek a TestImages alkönyvtárba kerülnek, egy „Image” alapnévvel. Amennyiben nem határozol meg útvonalat a kifejezésben (csak az alapnevet adod meg, ami a képek kivonásához szükséges), akkor a képek a PDF-fel azonos könyvtárba kerülnek. Én jobb szeretek alkönyvtárakat használni, csak a rend kedvéért.

Készítsünk képet az egyes PDF-oldalakról: pdftoppm

Ha képet szeretnél készíteni egy PDF-ről (vagy oldalairól), használhatod a **pdftoppm**-eszközt, szintén a Poppler fejlesztői jóvoltából. Miközben a pdfimages lehetővé teszi képek kiemelését a PDF-fájlból, a pdftoppm egy PDF teljes oldalairól képmásolat készítését teszi lehetővé.


From The Chief Editor's Desk...

A pdftoppm parancs formátuma hasonló a pdfimages-nél használthoz. Íme egy példa:

```
pdftoppm -jpeg Nov2022-Welcome.pdf Image
```

Ahogyan azt más parancsokkal is tettük, először cd paranccsal lépjünk be a könyvtárba, amiben az a PDF-fájl van, amiről a képet készítenénk. A -jpeg parancssori kapcsoló közli a pdftoppm-mel, hogy JPG-fájlokat készítsen. A -png kapcsoló PNG-fájlt készít, amíg a -tiff kapcsoló TIFF-eket készít. Kapcsolók nélkül a pdftoppm PPM-grafikus fájlokat készít.

Az „Image” paraméter adja meg, hogy mi legyen a készített képek nevének a töve. Sok más PDF parancssori eszközhöz hasonlóan opcionálisan megadhatod az oldaltartományt, amiről képek kell készíteni. Írd be **pdftoppm -help** a parancssorba, hogy lásd a lehetséges parancssori opciókat.

Ágyazzunk be, vagy szedjünk ki fájlokat a PDF-ből: pdfattach és pdfdetach

Itt van valami, amiről nem tudtam, hogy a PDF-fájlokkal megtehető. Jó … ez nem teljesen igaz. Tudtam, hogy képes ilyesmire, de eddig sosem próbáltam, vagy használtam ki ezt a képességét.

Te tudtad, hogy beilleszthetsz fájlokat PDF-fájlba, és később kiveheted onnan azokat? Történetesen ez SOKKAL könnyebb, mint gondolnád. Amikor rájöttem, valami ilyet éreztem: „Micsoda? Csak ennyi?”

Felhasználási a lehetőségei végtelenek. El sem tudom mondani, hányszor osztottam meg saját bash-szkriptet a magazinban, az olvasót elirányítva a Magazine weblapjára a szkript letöltéséhez. Sokszor kívántam azt, bárcsak lenne valamilyen mód arra, hogy a bash-szkriptet a felhasználónak a PDF-fel **együtt** juttassam el. Úgy tűnik, ez a lehetőség már régóta létezik.

Be lehet ágyazni egyszerű, könnyen olvasható fájlt a PDF-be, de lehetnek akár titkosított (azaz jelszóval védett tar.gz, vagy zip) fájlok is. Amennyiben az utóbbi esetről lenne szó, akkor győződj meg arról, hogy a kiszemelt címzett tudja, hogyan fejtse meg azokat a

fájlokat. A titkosított fájlok nyilvánvaló előnye, hogy ha valaki csak úgy belefut a PDF-edbe beágyazott fájlokba, nem tud hozzáférni az adatokhoz, hacsak nem tudja, hogyan fejtse vissza azokat a fájlokat. Tudom, ez mind egy kicsit rejtélyesen hangzik, de ez egy olyan lehetséges felállás, ami előfordulhat.

Ezzel szemben, titkosítatlan, könnyen olvasható fájlok beágyazása azzal kecsegtet, hogy a fájljaidat a lehető legtöbb emberrel megoszthatod. Könnyen beágyazhatsz egyszerre titkosított és nyílt fájlokat ugyanabba a PDF-be, lehetővé téve a fájlok kivonását, de csak néhány lesz elérhető mindenkinek, miközben a többi fájlhoz csak azok férhetnek hozzá, akik rendelkeznek a megoldó kulccsal.

Tehát, mindezeket figyelmbe véve nézzük meg, milyen egyszerű fájlokat beágyazni egy PDF-be és milyen egyszerű kiszedni fájlokat a PDF-ből. Természetesen, előbb be kell ágyazni fájlokat ahhoz, hogy legyen mit kivenni onnan. Ezek a parancsok is a Poppler fejlesztőinek köszönhetők.

Fájl PDF-fájlba beágyazásához a **pdfattach** parancsot kell használni. A parancs formátuma nem is lehetne ennél egyszerűbb.

(pdfattach <az eredeti fájl neve> <a beágyazandó fájl neve> <a kimeneti PDF neve>)

Ahogy a próbálkozásaimból tapasztalatai alapján elmondhatom, egyszerre csak egy fájlt lehet beágyazni az adott PDF-be. Tehát, ha több hozzáfűzendő fájlod van, egyenként, lépésenként kell csinálni. Ne aggódj, mivel nagyon gyors és könnyű. Könnyebbnek találtam belépni „cd”-vel az összes fájlt (a PDF-et és a mellékletek is) tartalmazó könyvtárba és onnan alkalmazni a műveletet. Ugyanakkor ***használhatsz*** teljes elérési útvonalakat is az egyes fájlok nevének megadásához, de ezzel több hibalehetőséget is teremtesz magadnak. Az összes fájlt egyetlen könyvtárba rakni, sokkal egyszerűbbé. Emellett én a parancs mindegyik változatához eltérő kimeneti fájlnevet adok. Ezáltal, **arra az esetre**, ha valami összekeveredne, nem kell mindent újra elkészíteni. Emellett sosem jó ötlet az eredeti fájlt felülírni. Ha úgy döntesz, később is átnevezheted.

Meglepő módon, fájlok beágyazása a PDF-be csak nagyon kis mérenövekedéssel jár. A végső méret becsléséhez az eredeti PDF-fájl bytokban megadott méretéhez add hozzá a hozzáfűzendő fájl byte-jainak számát és még további úgy 800 byte-ot. A 800 byte feltehetően az új PDF-ben a plusz, beágyazott fájl(ok)ra mutat belső hivatkozásokból következik. Ha egy pillanatra megállsz és belegondolsz, a 800 byte egyáltalán nem is túl sok plusz.

Most, hogy már a fájl bekerült a PDF-be, kell egy módarra, hogy azt kivegyük onnan. És ez az ahol a **pdfdetach** parancs jön a képbe. Miközben más opciói is vannak (az opciók megtekintéséhez írd be **pdfdetach –help**-et a parancssorba, vagy keress rá a neten a pdfdetach man page -re) de egyetlen nagyon jó módszer van arra, hogy a lehető legkönnyebben leválaszd azokat:

pdfdetach -saveall <name-of-PDF-file-with-attachments>

Amikor így futtatod, az összes PDF-be beágyazott fájlt leválasztja és a PDF-fel azonos könyvtárba írja ki. Ha egy kicsit elegánsabb szeretnél lenni, a beágyazott fájl(ok)nak útvonalat határozhatsz meg:

pdfdetach -o ./Files -saveall <name-of-PDF-file-with-attachments>
(padfdetach -o ./Files -saveall <a csatolt fájlokat tartalmazó PDF neve>)

Arról tudjál, hogy annak a könyvtárnak (./Files) már léteznie KELL, mivel a pdfdetach nem hozza létre neked a könyvtárat. Ha fájlt (vagy fájlokat) akarsz nemlétező könyvtárba kibontani, a parancs hibajelzést fog adni és egyetlen fájlt sem választ le. Ha sikeres, akkor az összes fájl a meghajtódra kiíródik, teljesen az eredeti fájlnévvel.

Rakjunk több oldalt egyetlen lapra: pdfxup

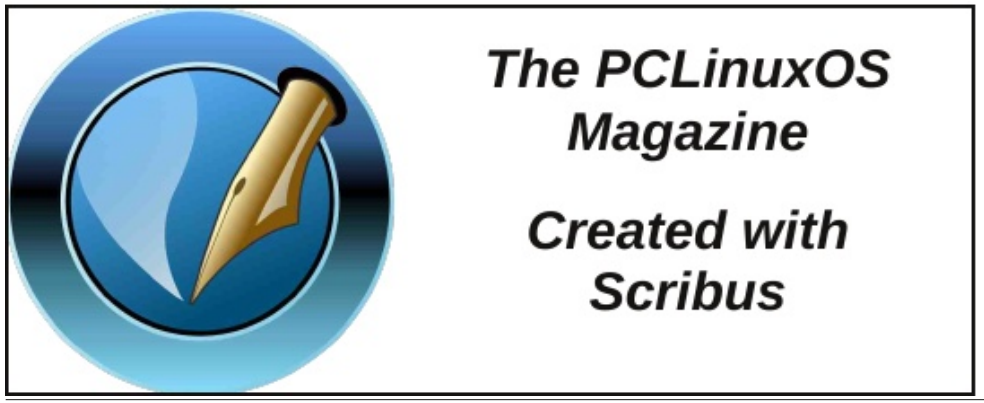
Néha előnyös lehet, ha képesek vagyunk egy PDF több oldalát egyetlen lapon megjeleníteni. Tudok ilyet, maikor részt veszek egy szemináriumon, és az előadó az előadása anyagát nyomtatásban, „lekicsinyített változatban szétosztja a hallgatóság között, gyakran három, négy, vagy éppen hat „diával” egy lapon. Ezzel lehetővé teszi, hogy kövesd az előadását és módot adarra is jegyzetek készíts az előadás mellé.

Vannak „más” programok, amik megcsinálják, hogy több oldalt egyetlen lapra feltesznek (ilyen a **pdfjam-slides3up** és a **pdfjam-slides6up**), de a **pdfxup** adja a legnagyobb szabadságot, miközben továbbra is nagyon könnyen használható marad. Azt azért vedd figyelembe, hogy e három program MINDEGYIKE eltávolítja a hivatkozásokat az eredeti PDF-ből, konverzió után ne keresd azokat.

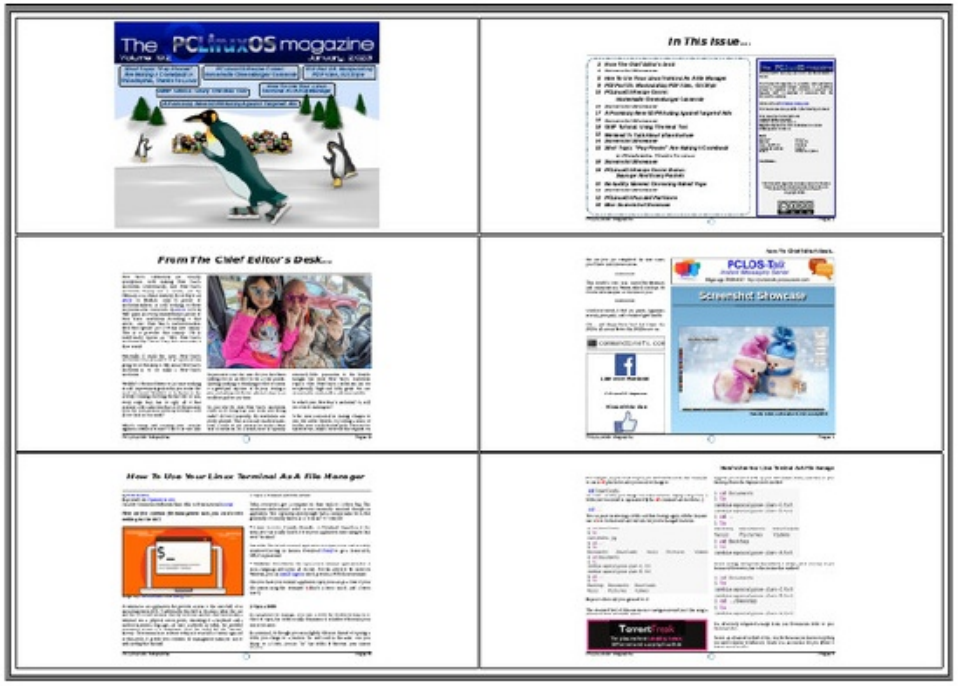
A pdfxup alapból az oldalakat álló formába rendezve készíti el. Íme a parancs és a kimenete:

```
[parnote-320@localhost PDF-3B]$ pdfxup -nup 2x3 -o 2023-01-
6up.pdf 2023-01.pdf
-> processing options
-> computing bounding
box.....................................
-> producing final file
final scale: 30.12543%
-> cleaning
```

A példámban a PCLinuxOS Magazine 2023. januári számát használtam fel. Ahogy ezen a cikken dolgoztam, cd paranccsal beléptem abba a könyvtárba, ahol a PDF-fájl van, hogy onnan futtassam a parancsot. Teljes útvonalleírásokkal is dolgozhatsz, de ez esetlenné és nehezen kezelhető teszi a parancsot, és sokkal nagyobb a hibázási lehetőséggel is jár. A -nup 2x3 mondja meg a pdfxup-nak, hogy az új PDF két oszlop széles és 3 sor mély legyen az egyes lapokon. Az -o paraméter a kimeneti új fájl számára meghatározza nevet. Új és egyedi nevet adjál, hogy elkerüld az eredeti fájlod felülírását. Ez esetben én hozzáfűztem egy „-6up” függeléket az eredeti névhez. A harmadik paraméter adja meg a használandó PDF-fájl, amiből a többoldalas PDF-et kell elkészíteni.



És íme, itt látható PDF-nézőben a folyamat eredménye:



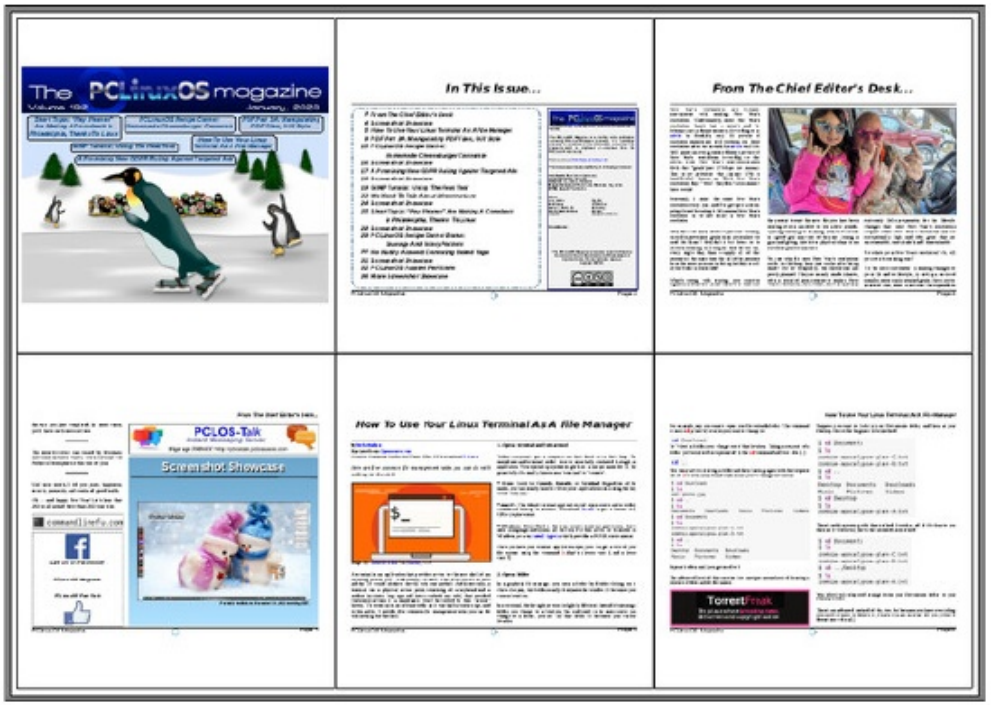
Amennyiben fekvő formátumú dokumentumot akarsz előállítani, egy kicsit módosítanod kell a parancsot. Itt van a parancs és a kimenete, ugyancsak a PCLinuxOS Magazine 2023. januári számát használva bemenetként:

```
[parnote-320@localhost PDF-3B]$ pdfxup -l -nup 3x2 -o 2023-
01-6up-l.pdf 2023-01.pdf
-> processing options
-> computing bounding box.....................................
-> producing final file
final scale: 33.69598%
-> cleaning
```

Először az -nup paramétert váltsd 2x3-ról 3x3-ra. Ezzel oldalanként három oszlop

szélességet és két sor magasságot érsz el. Add hozzá -l (kisbetűs „L”) paramétert, ami közli a pdfxup-pal, hogy fekvő tájolású új oldalakat készítsen.

Íme a parancs végeredménye:



Hiszed, vagy sem a többoldalas elrendezés olvasható marad. A PDF-nézőmben 500%-ra ki tudom nagyítani és kényelmesen olvashatom. Emellett még pont olyan jól néznek ki, mint a teljes méretű oldalak.

Ne válts többlapos változatra, ha abban a reményben tennéd, hogy kisebb PDF-fájlt eredményez. Semmi sem áll távolabb az igazságtól. Az eredmény nagyon nagy lesz. A PCLinuxOS magazine 2023. januári száma, amit a bemutatás céljára használtam, csak 7,1 MiB fájlméretű. Mindkét többoldalas PDF-elrendezés az eredeti PDF-nél körülbelül 6-szor nagyobb méretet ad, 43,3 MiB fájlmérettel… egyenként.

Semmi sem köt téged a 2x3-as, vagy 3x2-es elrendezéshez. Ha csak laponként két oldalt szeretnél, akkor legyen 1x2, vagy 2x1 attól függően, hogy álló, vagy fekvő tájolást szeretnél használni. De lehet akár 4x3, vagy éppen 3x4 … bármilyen kívánt elrendezés.

Keressünk a PDF-fájlokban: pdfgrep

Ha sok PDF-fájlt olvasol (és remélem, hogy legalább a The PCLinuxOS Magazine-t), eljön az az idő, amikor emlékszel, hogy olvastál valamit, de nehezedre esik előkeresni pontosan azt a PDF-fájlt, amiben az adott információ van.

A **pdfgrep** parancs segíthet megtalálni, ha valamit keresnél. Íme egy példa a parancs használatára:

```
pdfgrep -i pclinuxos 2023-01.pdf
```

Én általában cd paranccsal belépek abba a könyvtárba, amiben az átnézendő PDF-fájlok vannak. Az -i paraméter közli a pdfgrep-pel, hogy kis és nagybetűs is lehet az eredmény. A példában a keresett kifejezés a „pclinuxos”. Így a pdfgrep „találatot” jelez a „pclinuxos”-ra függetlenül attól, hogy kis-, vagy nagybetűs-e. A parancs végén pedig, hogy a The PCLinuxOS Magazine 2023. januári számát nézzük át.

Az átnézendő fájlok nevének meghatározásánál nyugodtan használhatsz dzsókert. Ha egy PDF-fájlokkal teli könyvtárad van, lecserélheted a „2023-01.pdf” fájlnevet *pdf-re, hogy a könyvtár összes PDF-fájlját átnézze. A pdfgrep kimenetét szövegfájlba át is irányíthatod, ha túl sok „találat” lenne keresési kifejezésre. Segíthet könnyebben megtalálni a keresett kifejezést. Többségünk nem képes olyan gyorsan olvasni mint, amilyen gyorsan lefutnak az adatok a terminál ablakában.

Vannak még további, használható parancssori kapcsolók is. Megnézheted a kínálatot a **pdfgrep --help** beírásával a parancssorba. Sokkal részletesebb opciós listáért próbáld meg beírni a **man pdfgrep**-et a parancssorba. Az opciók felfedezése rád van bízva. Az itt bemutatott pdfgrep csodás módon végignézi a pdf-fájlgyűjteményt, hogy előtalálja a keresett információt neked.

Legyünk furcsák: pdfflip

Őszintén bevallom, FOGALMAM SINCS miért akarná bárki is ezt csinálni, azon kívül, hogy vicces eredményt kap. Képzeld el, hogy egy félig átlátszó vetítővászon mögött állsz és nézed a vásznon éppen megjelenő dokumentumot. Ez az a látvány, amit a pdfflip-pel kapsz.

Íme a parancs:

```
pdfflip --suffix 'flip'
```

```
pdfflip --suffix 'flip' Nov2022-Welcome.pdf
```

És ez az amit a paranccsal kapsz:

Ahogy azt korábban említettem, FOGALMAM SINCS, hogy bárki is miért akarná ezt, kivéve szórakozásból, vagy azért mert képes erre. Tartsd észben, hogy miként az összes pdfjam eszköz futtatása esetén történik, a hivatkozások az átalakított PDF-fájlba nem kerülnek át. A pdfflip-eszköz nem működik a titkosított PDF-fájlokkal.

Összegzés

Ahogy láthatod, RENGETEG eszköz áll rendelkezésre PDF-ek manipulálására a parancssorban. Mindemellett, egyáltalán nem nehéz használni azokat. És az is vitán felül áll, hogy a feladatukat gyorsan hajtják végre.

Még több eszköz is van, amiket nem érintettünk, de úgy vélem azok vagy nagyon „speciális használatúak”, vagy az ismertetekkel azonos célúak. A legtöbb felhasználó PDF-manipulálási igényeinek az itt bemutatott parancsok több mint megfelelnek