

Általános fájl kezelő parancsok

touch

Létrehoz egy üres fájlt, vagy ha a fájl már létezik akkor pedig módosítja az utolsó hozzáférés és módosítás idejét egyidejűleg az aktuálisra.

Szintaktika:

`touch [-a | -m] [állománynév(ek)]`

- -a: csak a hozzáférési idő módosítása
- -m: csak a módosítási idő módosítása

cp

Fájl másolása egy helyről egy másik helyre.

Szintaktika:

`cp [forrásfájl] [célfájl]`

A fájl típusától függ, hogy mappán belül, egy másik mappába másol egy fájlt vagy egy mappát. Mappa másolás előtt ne felejtsek el, hogy külön kell beállítani, hogy rekurzívan működjön.

`cp -r [i] forrásmappa(k) célmappa(k)`

Ha a felülírás lehetőségét szeretnénk elkerülni, akkor használjuk az -i módosítót is, hogy interaktívan visszakérdezzen ilyen esetekben.

mv

Egy fájlt mozgat a könyvtárrendszerben egy helyről egy másikra vagy egy fájlt átnevez.

Szintaktika:

`mv [-i] [forrásfájl] [célfájl]`

Amennyiben a második argumentum fájlnev, akkor átnevezés történik, ha mappanév, akkor pedig átmozgatás. A forrásfájl lehet fájl is mappa is, a hatása ugyan az.

rm

Töröl egy vagy több fájlt. Vigyázat, mindent töröl, nem kérdez rá, ezért minden esetben toldjuk meg egy -i kapcsolóval, így rákérdez minden egyes elemre törlés előtt.

Szintaktika:

`rm [kapcsoló(k)] fájlnev...`

Kapcsolók:

- -f: kényszerített, hibajelzés elmaszkolása
- -i: interaktív
- -r|-R : rekurzív törlés

find

Megadott feltételeknek eleget tevő állományokat keres. A keresés nagyon erőforrás igényes és jelentősen leterheli a rendszert így mindig próbáljuk meg a keresési feltételeket leszűkíteni.

Szintaktika:

`find elérési_útvonal kifejezés [tevékenység]`

Kifejezések a keresendő fájlok megadásához:

Rengeteg kapcsolót tartalmaz (bővebben a manuálban találsz leírást)

- `-name név`: adott nevű fájlok keresése
- `-type fájl_típus`: Adott fájl_típusú fájlokat keres (pl: `d` - mappa)
- `-mtime [+/-]szam`: a legutolsó módosítás ideje napokban
- `-atime [+/-]szam`: a legutolsó hozzáférés ideje szintén napokban
- `-user userid`: melyik felhasználó tulajdonában van a fájl
- `-group csoportid`: melyik csoporté a fájl
- `-perm jogosultság`: hozzáférési jogosultság (3db oktális számjegy)
- `-size [+/-]szam[c]`: a megadott méretnél nagyobb vagy kisebb fájlok keresése (a méret blokkokban értendő, a `c` módosító esetén viszont bájtokban)
- `-a`: és kapcsolat a keresési feltételek között
- `-o`: vagy kapcsolat a keresési feltételek között

Tevékenység:

Találat esetén az adott fájlra végrehajtja a parancsot

- `-exec parancs { } \;`: ha találat van lefut a parancs
- `-ok parancs { } \;`: ha találat van lefut egy olyan parancs amely felhasználói inputot fog kérni
- `-ls`: listázza a talált fájlokat

Példa a find parancs használatára:

```
adamkoa@it:~$ find /hol/keresek -name valami* -a -size +256c -exec rm{ } \
```

Megkeres minden "valami"-vel kezdődő és 256 bájtól nagyobb állományt, majd törli azt.

Ahova nincs jogosultságunk belépni ott hibaüzeneted ad, ennek kiszűrésére toldjuk meg a `2>/dev/null` kapcsolóval, mely hatására a hibaüzenet nem jelenik meg a képernyőn (átirányítás a semmibe).

```
adamkoa@it:~$ find /hol/keresek -name valami* -ls 2>/dev/null
```

További példák:

```
find . -type d # könyvtárak keresése az aktuális mappában
```

```
find . -mtime +90 # amelyek nem lettek módosítva az elmúlt 90 napban
find ~ -perm 777 -a -size 400 # a home mappán belüli 400 blokknál nagyobb és
mindenki által módosítható fájlok keresése
```

cat

```
file      : Fájl tartalmát írja ki.
> file    : várja a bemenetet, amely a "file" tartalma lesz. Ctrl + D
kombinációval menthető.
-n file1: beszámozza a file1 sorait
??sh     : Minden .sh kiterjesztésű, 2 betűs file tartalmát kiírja a
képernyőre.

/dev/cdrom > /eleresi/utvonal/cd.iso      : A CD tartalmának ISO-ban
örténő mentése.
/etc/passwd |grep "/home" | cut -d: -f1 : A rendszerbe felvett
felhasználók kiírása

cat < bemenet.txt > kimenet.txt
# a cat beolvassa a bemenet.txt tartalmát és a kimenet.txt-be irányítja.

cat file.txt 1> file2.txt 2>&1
# A hibacsatorna is a kimenetre keverhető, azaz a file1.txt tartalma ÉS
a lehetséges hibák
# is bekerülnek a file2.txt-be. A hibacsatornáról a bash programozás
részben bővebben.
```

echo szoveg

```
# Kiírja a képernyőre a szoveg-et
echo szoveg > file      : a szoveg-et file-ba írja
echo $HOME              : $HOME nevű változó értékét adja meg, ami az
aktuális user home-ja. pl /home/letix
```

A standard perifériák és átirányításuk

Láttuk, hogy a programok túlnyomó többsége a billentyűzetről várja a bemeneti adatokat (input), és a képernyőre küldi az eredményeket (outputot).

E két perifériát, azaz a billentyűzetet és a képernyőt a UNIX standard be- és kimeneti csatornának nevezi, és kis egész számokkal lehet rájuk hivatkozni. Pontosabban fogalmazva, a UNIX három standard be- és kimeneti csatornát definiál, a *standard bemenetet* (**stdin, 0**); a *standard kimenetet* (**stdout, 1**) és a *standard hibacsatornát* (**stderr, 2**). A standard bemenet alapértelmezés szerint a billentyűzet, a kimenet és a hibacsatorna pedig a képernyő. Az utóbbi kettő szétválasztásának alapesetben nincs értelme, hiszen mind a futó programok eredményét, mind az esetleges hibaüzeneteket látni szeretnénk.

Nyilvánvaló, hogy nem sokra megyünk olyan programokkal, amelyek csak a billentyűzetről tudnak adatokat fogadni, és csak a képernyőre tudnak kiírni — e gondon segít az *átirányítás* (redirection). Ennek lényege az, hogy tetszőleges programot utasíthatunk arra, hogy bemenetét ne a billentyűzetről várja, illetve eredményeit ne a képernyőre írja. E mechanizmusnak a UNIX alatt kitüntetett jelentősége van: gyakorlatilag minden, a standard outputra író program kimenete átirányítható egy tetszőleges állományba, s hasonlóképp, bármelyik program, amelyik a standard

inputról olvas, tetszőleges állományból veheti inputját. A bemenet átirányításának jele a '<' karakter, a kimenet a '>' karakter.

Például az alábbi parancs a katalógus tartalmát nem a képernyőre listázza ki, hanem a file nevű állományba teszi:

```
$ ls -l >file
```

```
$ cat file
```

```
total 364
```

```
-rw-rw-rw- 1 demo guest 166262 Aug 23 20:29 FULL-INDEX
```

```
-rw-rw-rw- 1 demo guest 0 Aug 29 16:46 file
```

```
-rw-r-xr-x 1 janos guest 18 Aug 23 20:42 file1
```

```
drwxrwxrwx 3 demo guest 1024 Aug 23 20:59 newdir
```

```
..
```

```
-rw-rw-rw- 2 demo guest 18 Aug 23 20:59 text
```

```
$
```

A '<' és '>' karakterek mindkét oldalán tetszőleges számú szóköz karaktert tehetünk, de el is hagyhatjuk őket. Az alábbi parancssorozat először létrehozza az **akarmi** és a **semmi** nevű fájlokat, majd a **cat** parancs segítségével összemásolja őket egy harmadik, **temp** nevű állományba:

```
$ echo "Ez akarmi ***" >akarmi
```

```
$ cat akarmi
```

```
Ez akarmi ***
```

```
$ echo "**** Ez semmi" > semmi
```

```
$ cat semmi
```

```
**** Ez semmi
```

```
$ cat akarmi semmi>temp
```

```
$ cat temp
```

```
Ez akarmi ***
```

```
**** Ez semmi
```

```
$
```

A kimenet átirányításakor, ha az állomány, amibe az átirányítást eszközöltük, nem üres, akkor az újonnan beleíró adatok felülírják a tartalmát. Ennek elkerülésére szolgál a '>>' karakterekkel jelzett *additív átirányítás*; ekkor az átirányított adatok a meglevőkhöz fűződnek:

```
$ cat akarmi semmi >>temp
```

```
$ cat temp
```

```
Ez akarmi ***
```

```
*** Ez semmi
```

```
Ez akarmi ***
```

```
*** Ez semmi
```

```
$
```

A bemenet átirányítására egy példa: egy megírt levelet többeknek szeretnénk postázni. A levelezésre használt **mail** parancs elvileg interaktív működésű, azaz meg kell adni a parancssorban a címzett nevét és a levél tárgyát, majd a **RETURN** után begépelhetjük a levél szövegét. Ha viszont már készen lévő levelet szeretnénk küldeni, akkor a levelet a bemenet átirányításával adjuk meg a program számára:

```
$ mail -w janos geza <level
```

```
$
```

Az átirányítás segítségével lehetőség nyílik például a bejelentkezett felhasználók ábécésorrendben történő kilistázására:

```
$ who >temp; sort <temp
```

```
guest vt01 Mar 28 20:58
```

```
guest vt02 Mar 28 20:58
```

```
root console Mar 28 20:58
```




```
$
```
























A fentebbi példában használt **sort** parancs a bemenetről olvasott adatokat rendezi alfabetikusan, vagy számos más, opcionálisan beállítható szempont alapján.

Egy kis gyakorlás (amennyit bírsz megcsinálni)

Könyvtárkezelés

Kivonat: ls ~ (*, ? []), pwd, tree, cd, mkdir, rmdir, rm, mv, mc

- Melyik az aktuális könyvtár? 
- Lépjen a gyökérkönyvtárba! 
- Lépjen a saját home könyvtárába! 

- Lépjen a rendszergazda home könyvtárába (a jogosultsági rendszer valószínűleg megakadályozza majd)! 
- Lépjen a gyökérkönyvtárból nyíló etc könyvtárba! 
- Lépjen vissza egy szinttel feljebb! 
- Jelenítse meg az aktuális könyvtár tartalomjegyzékét! 
- Jelenítse meg a /etc, majd a /var/log könyvtár tartalomjegyzékét is (részletes adatokkal)! 
- Lépjen a saját home könyvtárába! Hozzon létre egy új alkönyvtárat, a neve legyen Teszt! 
- Egyetlen paranccsal hozzon létre ebben két újabb könyvtárat, T1-et és T2-t! 
- Rajzoltassa ki a könyvtárstruktúrát a tree paranccsal! A további feladatok megoldása során használja ezt a megoldások helyességének ellenőrzésére! 
- Egyetlen paranccsal hozzon létre a Teszt könyvtárból nyíló három, egymásból nyíló könyvtárat: Unix/Linux/Debian néven! 
- Nevezze át a Debian könyvtárat Deb-re! 
- Helyezze át a Deb könyvtárat a T1-be! 
- Törölje a T1 könyvtárat! 
- Egyetlen paranccsal törölje a Teszt könyvtárat! 
- Jelenítse meg az aktuális könyvtár tartalmát! 
- Jelenítse meg a /etc könyvtár tartalmát részletesen! 
- Jelenítse meg a /etc könyvtár conf kiterjesztésű fájljait! 
- Jelenítse meg a /etc könyvtár azon fájljait, melyek p-vel kezdődnek! 
- Jelenítse meg a /etc könyvtár azon fájljait, melyek f-re végződnek! 
- Jelenítse meg a /etc könyvtár azon fájljait, melyek első karaktere p, a harmadik s és d-re végződnek! 
- Jelenítse meg a home könyvtárának tartalmát a benne levő rejtett fájlokkal együtt! 
- Jelenítse meg a /etc könyvtár azon fájljait, melyek második karaktere a vagy n! 
- Jelenítse meg a /etc könyvtár azon fájljait, melyek második karaktere nem a és nem n! 
- Keresse meg a rendszer összes .conf kiterjesztésű fájlját (hosszan fut)! 
- Indítsa el a Midnight Commanert! 
- A fenti feladatokat végezze el a Midnight Commanderrel is!













Fájlkezelő parancsok

Kivonat: touch, cp, mv, rm, cat, file, TODO: touch.

gyakorlat

|

```
| --Proba
| |
| | --test
| | | ---unix
| |
| | --Linux
| |
| --Check
| | --Tmp
```

- Hozza létre a home könyvtárában az ábrán látható könyvtárakat! 
- Lépjen a /etc könyvtárba! Jelenítse meg az innen nyíló könyvtárrendszert a tree paranccsal! Csak a könyvtárakat jelenítse meg, a fájlokat ne! 
- Másolja a /etc könyvtár passwd fájlját a Proba könyvtárba! 
- Másolja a /etc könyvtár shadow nevű fájlját a test könyvtárba! 
- Másolja a /etc könyvtár összes conf kiterjesztésű fájlját a Linux könyvtárba! 
- Másolja a /etc könyvtár azon fájljait, melynek második karaktere a, a negyedik s, és d-re végződnek, a Check könyvtárba! 
- Helyezze át a Test könyvtár minden fájlját a Linux könyvtárba! 
- Hozzon létre egy új fájlt a Check könyvtárban, melynek neve Adatok.txt! 
- Törölje a Linux könyvtár fájljait! 
- Törölje a Próba könyvtárat! 
- Jelenítse meg a képernyőn a /etc/passwd fájl tartalmát! 
- Az előző feladatot végezze el a less paranccsal is! Értelmezze a fájl szerkezetét! * Lapozzon a szövegben, keresse meg a Bela nevű felhasználókat! Lépjen ki a less-ből! 
- Állapítsa meg a következő fájlok típusát: /bin/bash, /etc/passwd, /var/log/syslog! 