Karakteres munkamenet-kezelő

Adam a Screen sokféle előnyét és használati módját mutatja be.

A Screen olyan terminálnyaláboló (terminal multiplexer) program, amely sok folyamat kezelését teszi lehetővé egyetlen terminálon keresztül. Minden egyes folyamat saját virtuális ablakot kap, és a folyamatokkal kölcsönhatásban lévő virtuális ablakokat szabadon váltogathatjuk. A Screen által kezelt folyamatok akkor is tovább futnak, ha az ablakuk éppen nem működik.
Mindeddig tehát a Screen által nyújtott szolgáltatások nem túl érdekesek, de még csak nem is újak. Voltaképpen már léteznek olyan X11 terminálalkalmazások, amelyek képesek biztosítani ezt a szolgáltatást: a `konsole` és a `multi-gnome-terminal`. A Screent a többi programtól néhány kulcsszolgáltatás különbözteti meg.
A Screen képes leválni egy munkamenetről (session), és ahhoz egy későbbi időpontban újra csatlakozni. A munkamenetről történő leválasztás után a folyamatokat figyelő Screen továbbra is fut. Amennyiben a későbbiek folyamán újra csatlakozol a munkamenethez, termináljaid ugyanúgy várnak rád, ahogyan otthagytad őket.
A Screen minden egyes általa kezelt ablak számára egyedi, visszakereshető és lapozható vagy görgethető (scrollback) tárterületeket (puffereket) tart fenn. Megadhatunk „add meg a kívánt szót, és én megkeresem neked” típusú keresési parancsokat, de növekményes keresési műveleteket is végezhetünk. Ez annyira magától értetődő szolgáltatás, hogy már szinte az a meglepő, hogy több terminálemulátor nem ajánlja fel.
A Screen említésre méltó szolgáltatásai között megtalálhatók a beállítható billentyű-hozzárendelések, az `utf8` és többféle karakterkészlet-támogatás, többszörös csatlakozás (multi-attach), beállítható bementi és kimeneti fordítás, bemeneti és kimeneti szűrő hozzáférés-szabályozó listákkal (Access Control List – ACL), több felhasználó támogatása, valamint naplózás.

Kapcsolatban a Screennel

Mielőtt ténylegesen futtatnánk a Screent, fontos megértenünk, miként kell vele együttműködnünk. A Screen a vezérlőkarakter kivételével az összes bevitt szöveget elküldi a pillanatnyilag működő ablaknak.
Az előre beállított vezérlőkarakter a CTRL-A – tartsuk lenyomva a CTRL billentyűt, miközben leütjük az A billentyűt. A Screen súgóoldal, az Emacshoz hasonló módon, a CTRL jelölésére a C billentyűt használja.
A vezérlőkarakter arra használható, hogy tudassuk a Screennel, ezúttal magának a Screen programnak a működését szeretnénk szabályozni, nem pedig az adott ablakban megjelenő alkalmazását.
A vezérlőkarakter után lenyomott billentyű jelzi, hogy melyik Screen parancsot szeretnénk végrehajtani. Néhány lényeges parancsot és az azokhoz tartozó billentyű-hozzárendeléseket *táblázatunkban* foglaltuk össze. Sok gyakran használt parancsnál a billentyűműködés-szabályozó változata is parancshoz van kötve. Példaként említhetjük az ablak létrehozásához használatos CTRL-A C és a CTRL-A CTRL-C billentyűket. Ahhoz viszont, hogy a CTRL-A kombinációt egy alkalmazáshoz eljuttathassuk, a CTRL-A A kombinációt kell használnunk.
Ha szükséges, a vezérlőkarakter más karakterre cserélhető. Az Emacs-felhasználók a vezérlőkaraktert jellemzően CTRL-B-re szokták cserélni oly módon, hogy a ***.screenrc*** állományukat az `escape Bb` sorral egészítik ki. Valamennyi alábbi példánkban viszont a CTRL-A billentyűegyüttest használjuk, mert ez az alapértelmezés.
Amint az várható, a ***.screenrc*** az adott felhasználóhoz tartozó saját könyvtárban (***/home***) a felhasználókra külön-külön jellemző saját beállításokat tartalmazza, miközben a ***/etc/screenrc*** állomány az egész rendszer működését érintő beállítóállomány, amely az összes felhasználóra érvényes.

Ablakkezelési alapok

Most, hogy már tisztáztuk a Screennel való „együttműködés” alapjait, végigfuthatunk a Screen-alapismereteken, példaként véve egy jellemző Screen-munkamenetet. A Screennek valószínűleg legjellemzőbb használata az olyan távoli géphez csatlakozó terminálok irányítása, amelyekre felhasználói azonosítóval rendelkezünk.
Azok, akik eddig csak játékkal töltötték otthon az időt, most jelentkezzenek be egy távoli gépre, amelyre már telepítették a Screent. Ha nem tudunk olyan gépről, ahová `ssh`-val be lehetne lépni, akkor telepítsd a Screent és `ssh`-val lépj be a helyi gépre (localhost).
A Screen-csomagok a legtöbb Linux-változat számára elérhetők. Most tehát előttünk van a távoli gép parancssora.
A parancssorba gépeljük be a `screen` szót. Ekkor a bejelentkező-képernyőnek kell megjelennie, amelyen tájékoztató adatok láthatók arról, hogy a Screen a GPL szabályozás alá tartozik, valamint azt, hogy hol lehet jelezni az esetlegesen felfedezett hibákat és más ehhez hasonló adatokat. A bemutatkozó képernyőt a szóközbillentyű lenyomásával hagyhatjuk el, de a ***.screenrc*** állományban a `startup_message` változó értékének `off`-ra állításával végérvényesen ki is kapcsolhatjuk. Ezután újabb parancssorhoz kell érkeznünk, ámde ez már a Screen programon belül fut.
A Screen programon belüli héjprogramnak ugyanúgy kell működnie, ahogyan az első tenné. Ha kiadod a `printenv` parancsot, akkor láthatod, hogy néhány új környezeti változó is be lett állítva. A Screen program a `TERM` változó értékét `screen`-re állítja, minden egyes ablak vt100-megfelelő virtuális terminált biztosít. A `WINDOW` változó értéke a virtuális ablakszámra, az `STY` változó pedig a munkamenet (session) nevére lesz állítva. Az utóbbi két fogalmat a későbbiekben fogom elmagyarázni.
Mindeddig ez az egyszerű rendszerhéj (shell) pontosan úgy működik, ahogyan azt egy távoli gépen már megszokhattad. A példa kedvéért kezdjük a Screen legfrissebb változatának – ez az írás létrejöttekor a 3.9.13 volt – letöltésével a Screen program terjesztési helyéről, a ➲ ftp.uni-erlangen.de/pub/utilities/screen címről. Az állomány letöltődése közben, az üres idő hasznosítása végett, dönthetsz a saját könyvtárad

A parancsok és a hozzájuk kapcsolódó billentyűkombinációk

A parancs neve	A billentyű	Az általa végzett művelet leírása
screen	C	Új ablak létrehozása.
kill	K	Ablak bezárása.
help	?	A választható parancsok és billentyűkombinációk listázása.
detach	D	Leválasztás a screen-munkamenetről.
next	N	A következő ablak kijelölése.
prev	P	Az előző ablak kijelölése.
other	CTRL-A	Átváltás az előző ablakra.
windowlist	"	Az összes screen-ablak megjelenítése. Ez a billentyű lehetővé teszi a lista egyik elemének a kiválasztását.
copy	[	Átkapcsolás, másolás és beillesztés, illetve visszalapozás üzemmódba.
break	B	Megszakítás (break) küldése.
lockscreen	X	A pillanatnyi munkamenet bezárása.
kettős pont	:	A screen-parancssor elérése. (A kettős pont használata hasonló a vi szerkesztőben megszokotthoz).
Select	'	Megadhatjuk a kívánt munkamenet számát vagy nevét, amire váltani szeretnénk.
windowlist -b	"	Kilistázza az összes munkamenetet, a nyílbillentyűkkel választhatunk közülük.
select 0	0-9	0 től 9-ig vált a munkamenetek között.
title	A	A munkamenet nevét adhatjuk meg.
clear	C	Kipucolja a képernyőt.

(*/home*) kitakarítása mellett. Ha nem használnánk a Screent, akkor egy újabb `xterm`-ablakot kellene nyitnunk, és `ssh`-val be kell lépnünk a távoli gépre. A Screen programban viszont a CTRL-A, C billentyűk új ablakot hoznak létre, új héjfolyamattal.
Eddig tehát rendelkezünk egy FTP-ügyféllel meg egy héjprogrammal, amelyik szorgalmasan tisztogatja a saját könyvtárat. Közben a CTRL-A, P billentyűkkel az eredeti FTP-ablakban ellenőrizhetjük a letöltést – ezzel a paranccsal az előző ablakba jutunk vissza.
A héjprogramhoz a CTRL-A, N billentyűkkel térhetünk vissza. Ellenőrizzük a letöltést, hogy befejeződött-e már. A Screen letöltése egyébként nem vesz olyan sok időt igénybe, de a példa kedvéért most úgy teszünk, mintha mégis sokáig tartana. Ugye, most már ideje lenne visszatérni a kacatokkal telezsúfolt saját könyvtárhoz? Mielőtt ezt megtennénk, nyomjuk le a CTRL-A, M billentyűket, hogy figyelemmel kísérhessük az éppen működő ablak kimenetét. A Screen most értesítést fog küldeni, ha az FTP-ablakban valamilyen tevékenység zajlik. A letöltési folyamat ellenőrzéséhez ezentúl már nincs szükség az ablakok váltogatására. Sőt ez fordított esetben is működik: használjuk a CTRL-A, _ billentyűket, hogy a tevékenység nélküli időszakot figyelemmel kísérhessük, ez alapértelmezés szerint 30 másodperc. A tevékenység nélküli időszak megfigyelése különösen hasznos hosszú programfordításoknál vagy más olyan munkáknál, amelyek ontják az adatokat.
Sorozatban további héjprogramokat tartalmazó ablakokat hozhatunk létre, és ezzel párhuzamosan végezhetjük tennivalóinkat a távoli gépen. Néhány ablak létrehozása után nehéz nyomon követni, hogy melyik ablak hol is található. Ezen a ponton jut szerephez a `windowlist`. Nyomjuk meg a CTRL-A," billentyűket, mire megjelenik az éppen megnyitott ablakokat tartalmazó lista. A listában a J és a K billentyűkkel lehet mozogni. Amikor a lista kijelölt bejegyzésén állva leütjük az ENTER billentyűt, az válik a pillanatnyi ablakká. Az előre beállított értéknek megfelelően az ablaknév nem valami beszédes. Ezen a helyzeten úgy lehet javítani, hogy az ablak nevét a CTRL-A, SHIFT-A billentyűkombinációkkal magunk állítjuk be. Ezeket a címeket önműködően is be lehet állítani, akárcsak az `xterm`-ben tennénk: egy ESC-K, majd a cím, végül az ESC-\ jelsorozat elküldésével.
Nagyon valószínű, hogy magad is képes leszel alkalmazni a héjprogramnak jobban megfelelő receptet, hogy a megadott átváltási sorozattal (escape sequence) az `xterm` címsorát a `screen` ablak nevére változtasd.
Az ablakkezeléssel rohamléptekben zajló ismerkedésünk befejezéseként zárjuk be az összes ablakot. Ha kilépsz a Screen által gyártott héjprogramból, egyúttal az ablak is önműködően törlődik. Az ablak a `kill` parancs segítségével kézzel is törölhető, az alapértelmezés a CTRL-A, K. Ha az összes Screen ablakból kilépünk, a Screen is befejezi működését. A Screen a `quit` paranccsal arra is utasítható, hogy zárja be az összes ablakot, és lépjen ki – ezt a CTRL-A, \ billentyűkkel lehet gyorsan megtenni.

Munkamenetek

Ha egymás után hozol létre új ablakokat és váltogatod őket, lehetőséged nyílik rá, hogy teendőidet párhuzamosan végezd. Használhatsz mondjuk egy-két szövegszerkesztőt, IRC-csevegőt és még jó néhány további programot, mindegyiket a saját ablakában. Az esetenként előforduló katasztrófa azonban mindig váratlanul történik, és a hálózati kapcsolat megszakad. Akik még mindig játszottak az otthoni gépükön, most aztán bezárhatják SSH ügyfélprogramjukat. Ugye, úgy tűnik, ideje elkezdeni összeszedni a rendszer darabjait, és újra elindítani az alkalmazásokat a távoli gépen? Nos, ha Screent használunk, akkor erről szó sincs.
Minden alkalommal, amikor a Screent kapcsolók nélkül indítjuk el, az új munkamenetet hoz létre. Ez két folyamatot kelt életre: egy terminálkezelő folyamatot és egy ügyfélfolyamatot. Az ügyfélfolyamat önműködően hozzá van kapcsolva a terminálkezelési folyamathoz. Amikor tehát gépelünk, a begépelt karakterek az ügyfélprogramhoz kerülnek, ami a megfelelő alkalmazáshoz küldi őket tovább.
Abban az esetben, ha a hálózati kapcsolatod megszakad, az ügyfélprogram veszi a jelet, és leválik a terminálkezelési folyamatról. A terminálkezelési folyamat tovább dolgozik, figyelemmel kísérve a terminálokat, mintha mi sem történt volna. Amikor újra visszajelentkezel a rendszerbe, a parancssorban kiadott `screen -ls` paranccsal listát készíthetünk a futó munkamenetekről. Ennek az alábbi üzenethez hasonlót kell eredményeznie:

```
There are screens on:
          24319.pts-9.hostname (Detached)
1 Sockets in /var/run/screen/S-userid.
```

Gnome-Multi-Terminal

A cikkben a szerző is említést tesz a Gnome grafikus munkafolyamat-kezelőről, ami tulajdonképpen ugyanazt hivatott ellátni, mint a `screen`, csak grafikus elemekkel van bővítve. Ugyanúgy új terminálablakokat nyithatunk, amelyeket itt különféle fülek jeleznek, és a közöttük történő váltást is a segítségükkel tehetjük meg. A beállítás menüben a felületet testreszabhatjuk, vagyis hogy mi jelenjen meg és mi ne az ablak keretén belül. Választhatunk a *Toolbar* vagy a *Buttonbar* között; mindkettőt nem érdemes használni, mivel ugyanazokat a lehetőségeket kínálják. A *Toolbar* nagyobb helyigényű, mint a *Buttonbar*, így egyértelműen az utóbbi mellett szavazok. Az én `screen`-emnél tapasztalt véletlenszerű lefagyások miatt kénytelen voltam átszokni a `gnome-multi-terminal`-ra és a grafikus felületre, ami most már nem zavar annyira, mint az elején. Sokszor nyomkodtam a megszokott billentyűket, megszokásból, sajnos ezeket itt nem tudtam használni. Ennél is lecserélhetjük az egyes folyamatokhoz tartozó neveket valami sokatmondóbbra, tehát nem kell tíz *Shell* nevű ablak között habozni, hanem különféle pontos meghatározásokat írhatunk hozzájuk (például `tail -f syslog`).

Csontos Gyula (Csontos.Gyula@linuxvilag.hu)

Ez az üzenet azt mutatja, hogy a munkamenet önműködően vált le, amikor a kapcsolat lebontott.
A munkamenethez való újrakapcsolódásra többféle lehetőség is kínálkozik. Egyértelműen meghatározhatjuk a munkamenet nevét:

```
# screen -r munkamenetnev
```

A munkamenet számára előírhatod, hogy csatlakozzon újra, ha ez lehetséges, különben pedig kezdjen új munkamenetet:

```
# screen -R
```

Választhatjuk még a „tedd, amit kell, csak megkapjam a Screen-gépemet” elméletet, és használjuk a `screen -D -RR` parancsot. Eme legutóbbi lehetőség a már csatlakozott ügyfeleket is leválasztja, és a felsorolásban szereplő első munkamenethez rendeli hozzá.
Amikor ezen parancsok valamelyikét használod, tudnod kell, hol hagytad abba a munkát, még mielőtt még hálózati kapcsolat tönkrement volna. Az újracsatlakozás után a munka úgy folytatható, mintha mi sem történt volna.
Egy munkamenethez való többszöri csatlakozásra is adott a lehetőség. Ennek akkor vesszük hasznát, ha a Screen-munkamenetet egy másik gépről még nem zártuk le, vagy csak egyszerűen ugyanazonabból a munkamenetből szeretnénk ablakokat egymás mellé sorakoztatni. A többszörös csatlakozáshoz a parancssorban a `screen` parancs kiadásakor használjuk a `-x` kapcsolót.
Végül, amikor közeleg a munkanap vége, és elérkezik a hazaindulás ideje, a munkamenetről való leválás a CTRL-A, D parancs használatával érhető el. Másnap, amikor újra csatlakozol, ugyanott leszel, ahol a munkát abbahagytad.

A másolás és beillesztés, illetve a visszalapozás üzemmód

A Screennek írásunk elején említett egyik kulcsfontosságú szolgáltatása a visszakereshető visszalapozás (searchable scrollback). Ez egy olyan szolgáltatás, amelyet sohasem tudtam nélkülözni. Az újdonsült Screen-felhasználók számára ez nem értetődik azonnal magától, de a Screen visszalapozása a másolás parancson keresztül érhető el. Másolás módba a CTRL-A, [billentyűkkel vagy a `copy` paranccsal lehet kerülni. A navigáció a várakozásoknak megfelelően működik: használhatók a kurzormozgató nyilak, a PageUp PageDown lapozóbillentyűk, vagy a `vi` szerkesztőprogramban megszokott megfelelőik.
A `vi`-stílusú kereséshez a / (perjel) vagy ? (kérdőjelek) használandók, növekményes kereséshez a CTRL-S, illetve a CTRL-R billentyűkombinációk. A kis- és nagybetűk megkülönböztetése az `ignorecase yes` paranccsal kapcsolható ki.
Amennyiben a másolásmódot kizárólag a visszalapozáshoz használjuk, akkor ebből az ESC billentyű lenyomásával bármikor kiléphetünk.
Szöveg másolásához először igazítsuk a kurzort (a kurzort helyőrnek is hívják – a szerk.) a másolandó szöveg elejéhez, majd a kijelöléshez üssük le a szóközbillentyűt. Ezután vigyük a kurzort a másolni kívánt szöveg végéhez, és a kijelölés befejezéséhez ismét nyomjuk le a szóközbillentyűt. Amint a másolandó szöveg végét kijelöljük, a szöveg a másoláshoz használt belső ideiglenes tárterületbe kerül, egyúttal a másolási üzemmódból is kilépünk. Utána ebből az átmeneti tárterületből a szöveget a CTRL-A] billentyűk segítségével a működő ablakba másolhatjuk.
Az utolsó jellemző, amelyet a másolás–beillesztés módról tudnunk kell, az a visszalapozási átmeneti tár, ami az előre beállított érték szerint száz sorra terjed ki. Véleményem szerint ez nem elegendő. Az értéket magasabbra állíthatjuk, ha a ***.screenrc*** állományba beszúrjuk a `defscrollback 1024` parancsot tartalmazó sort.

Ablakok megnyitása a .screenrc-vel

Mint már fentebb említettem, a ***.screenrc*** állományt parancsokkal egészíthetjük ki, hogy Screen viselkedését a saját igényeinkhez igazítsuk. Nem magától értetődő, hogy a `screenrc`-be bármilyen Screen-parancs beírható. Ez roppant hasznos, és az ablakgyártást (spawn) rögtön a `screen` parancs kiadásakor önműködően meg lehet kezdeni.
E csemegének számító ismeret alkalmazása a Screennek előre meghatározott ablakkészlettel való indítása. A következőkben egy éppen ilyen feladatokat ellátó minta ***.screenrc*** állományt láthatunk:

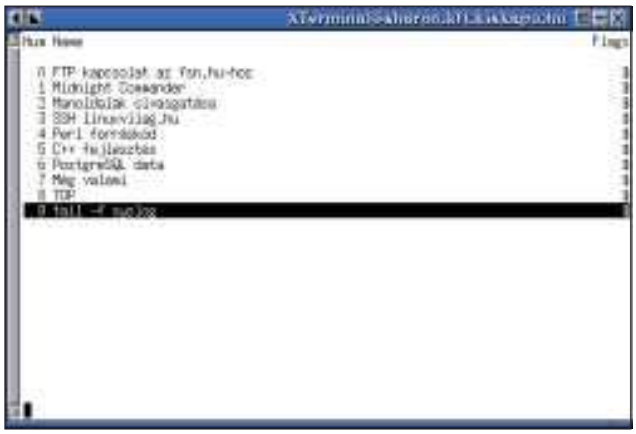
```
# elısz r olvassuk be a sajÆt screenrc
# ÆllomÆnyunkat mielıtt bÆrmi mÆst tesz nk
source $HOME/.screenrc

# most pedig kezdj k el az ablakok kinyitÆsÆt
screen top

# az ablak c msorÆban lØvı c met
# megvÆltoztathatjuk a -t lehetısØg rØvØn
# megvÆltoztathatjuk
screen -t irc epic

# meghatÆrozhatjuk az ablak szÆmÆt,
# melyikben induljon az alkalmazÆs
screen -t mail 8 mutt
```

```
screen -t daemon 9 tail -f /var/log/daemon.log
```

Ha ezeket a parancsokat a ***$HOME/.screenrc.multiwin*** állományba mentjük, a

```
# screen -c $HOME/.screenrc.multiwin
```

paranccsal rávehetjük a Screent, hogy a saját könyvtárban található ***screenrc*** állományunk helyett ezt használja.



Az indító héjprogramokból olyan Screen-munkamenetek is indíthatók, amelyek inkább a rendszer jellemzőit mutatják. A rendszer Screen-munkamenet gyakori alkalmazása a soros konzolkiszolgáló. A Screen kitűnő erre a célra, minthogy a soros terminálokhoz beépített támogatással és naplózással van felszerelve. Az alábbiakban egy ilyen célt szolgáló, megjegyzésekkel ellátott ***screenrc*** állományt mutatunk be:

```
# Ennek az ÆllomÆnynak a hasznÆlata azt
# elıfeltØtelezi, hogy a felhasznÆl
# rendelkezik a megfelelı jogosultsÆgokkal,
# hogy a soros kapukat hasznÆlja Øs a
# napl ÆllomÆnyokba  rjon


# kapcsoljuk be a napl zÆst az  sszes ablakra
# vonatkoz an
deflog on

# a screen napl ÆllomÆnya a
# /var/log/serial.$WINDOW legyen
logfile /var/log/serial.$WINDOW

# ablakok nyitÆsa a soros kapun
screen /dev/ttyS0 38400
screen /dev/ttyS1 19200
```

Ha ezeket a parancsokat a */etc/screenrc.serial* állományba mentjük, ezt a következő héjprogrammal lehet a Screen indulásakor bekapcsolni:

```
# su serialuser -c screen -dms serial -c
➥/etc/screen.serial
```

A `-dms serial` kapcsoló arra utasítja a Screent, hogy a munkamenetet leválasztott üzemmódban indítsa el, és „serial”-nak nevezze el. A soros kapu használatához jogosultsággal rendelkező felhasználó bejelentkezhet és csatlakozhat ehhez a munkamenethez, pontosan úgy, mint bármilyen másik közönséges Screen-munkamenethez. A Screen leválasztott módú indítása lehetővé teszi, hogy a Screent `cron` feladatként indítsuk el, amennyiben éppen erre van szükségünk. Lehetséges egyetlen, az egész rendszerre érvényes ***screenrc*** létrehozása, amelyhez több felhasználó csatlakozhat. A Screen az ablakokhoz kapcsolódó ACL-ek (Access Control List) révén támogatja a többfelhasználós üzemmódot, vagyis megszabja, hogy melyik felhasználó mit tehet és mit nem. A több felhasználó által alkalmazható Screen azonban `setuid root` üzemmódot igényel. Eme követelmény miatt többfelhasználós Screen-munkameneteket bemutató példákat szerintem nem célszerű taglalni. Ha mégis többfelhasználós Screen-munkamenetet szeretnél üzembe állítani, akkor alaposan tanulmányozd át a Screen leírását, tedd fel a „`setuid root` jogosultságok adása egy bonyolult programkódhoz” üldözési mániás kalapodat, és készülj fel a lehetőségek lehető legszorosabb korlátozására. A harmadik megoldás az, hogy az előző két példát összefésülöd, és az egész rendszert érintő felhasználói közreműködést igénylő (interaktív) programokat a ***screenrc***-n keresztül indítod el. Ennek remek kihasználása a Mutella – a konzolalapú Gnutella ügyfélprogram – elindítása a Screen indulásakor. A Screennel ezt el lehet indítani, alkalmanként kapcsolódni lehet hozzá, meg lehet tudni az állapotát, és lekérdezéseket is lehet vele futtatni.

A screen program a 44. CD Magazin/screen könyvtárában található.

Linux Journal 2003. január, 105. szám

Adam Lazur (adam@lazur.org)
Linux-szaktanácsadó, működési területe a beágyazott rendszerektől a Beowulf-telepekig terjed. Szabadidejében szeret magáról egyes szám harmadik személyben írogatni. Adam szívesen fogad a cikkel kapcsolatos leveleket.

KAPCSOLÓDÓ CÍMEK

- http://www.math.fu-berlin.de/~guckes/screen
- http://groups.yahoo.com/group/gnu-screen