Rendszerfigyelés Conky-val, 1. rész

PCLinuxOS Magazine – 2015. június

Írta: Peter Kelly (critter)

„Conky ingyenes, könnyűsúlyú rendszerfigyelő X-hez, ami a képernyőn jelenít meg bármilyen információt.” A leírás a hivatalos weblapról való és eléggé elbagatellizálja ennek a hasznos kis alkalmazásnak a képességeit. A névről a weblap elmondja, hogy „a név a »Trailer Park Boys« kanadai TV-sorozat szereplőjétől való”, ahogy mondják egy hasbeszélő bábú. De ne hagyd, hogy ez megtévesszen. Conky egy komoly és hatékony alkalmazás. Az ötletet egy „TORSMO” nevű hasonló eszköztől vették át, a rövidítés a Tyopoyta ORvelo System Monitor-t takarja, aminek biztosan van valami jelentése és az alkotója biztosan megfelelőnek találta. Conky a PCLinuxOS tárolóiban elérhető.

Mára a Conky egy érett programmá vált rengeteg beállítási lehetőséggel és hozzáféréssel a rendszer számos jellemzőjéhez, mint a memóriahasználat és hálózati forgalom, mint alap. Emellett a Conky héjfüggvényeket képes futtatni, az eredményt a képernyőre rakva. Gondoskodtak if-then-else típusú feltételes kimenetről is és a tárolóban lévő legfrissebb verzió a Lua leíró nyelvhez is rendelkezik hozzárendeléssel. Utóbbi lehetővé teszi a vállalkozószelleműbb felhasználóknak, hogy kamatoztassanak néhány fejlettebb tulajdonságot, mint a Cairo grafikus könyvtárak használatát. Egy a felhasználók körben népszerűnek tűnő tulajdonsága, hogy a Conky a kimenetet áttetsző háttérrel jeleníti meg az asztali tapétán látható módon.

A Conky népszerűségének köszönhetően példák és letölthető szkriptek özöne érhető el az Interneten, lehető téve, hogy a kevésbé ambiciózus felhasználó is élvezhesse ezen kiterjesztett tulajdonságok egy részét.

A Conky nem korlátozódik unalmas rendszerstatisztikák megjelenítésére. Mutathatod vele az időt, képeket, időjárás-előrejelzést, vagy bármit, amit csak képzeleted kíván. Néhány haladó példa itt látható: a http://www.deviantart.com/browse/all/?q=conky. A weblap sok példája mellett ott van a hozzá tartozó és a gépeden használatra letölthető kód. Ezek közül sok nagyon fejlett technika megoldás, ami az alap Conky-ban nem érhető el és a cikk második részében fogunk majd foglalkozni velük. Addigra remélhetően elég sok elméleti kérdést tisztázunk, ami képessé tesz azok megértésére is.

Ez a cikk néhány kevésbé bonyolult conky példa bemutatásával indít, végigvezetve téged a Conky képernyőd beállításának alapjain, illetve bemutatja, hogyan lehet kimenetet létrehozni héj-parancsok használatával. A második rész bemutatja hogyan lehet a Lua szkriptekhez kapcsolódni, és felhasználni a Cairo könyvtárak függvényeit.

Első lépések

Indításként bemutatom lépésről lépésre hogyan készítsünk hasznos rendszerfigyelőt az általam használt alapján – jobbra áttetszőség nélkül látható. A fájl eredetijét valahonnan erről a fórumról szedtem össze valamikor régen, de sajnos nem emlékszem, hogy ki rakta fel. Bárki is volt, köszönet érte! Ugyanakkor a tartalmát elég jelentősen átdolgoztam, hogy megfeleljen a saját igényeimnek.

A Conky beállító fájlja sima, kétrészes szövegfájl, mindkettő szigorúan kötött formát követve. Az első részben a beállítások találhatóak, amik meghatározzák, hogyan és hol jelenjen meg a Conky ablaka és tartalma. A második rész egy önálló sorban álló „TEXT” szó után kezdődik. Minden ezután a szó után megjelenítésre szánt. Mondom „szánt”, mivel csak akkor jelenik meg megfelelően, ha a formátuma hibátlan. A Conky PCLinuxOS tárolóból telepítését követően nyiss terminált és írd be a **conky** szót. A bal oldalt mutatott ablakot kell látnod.

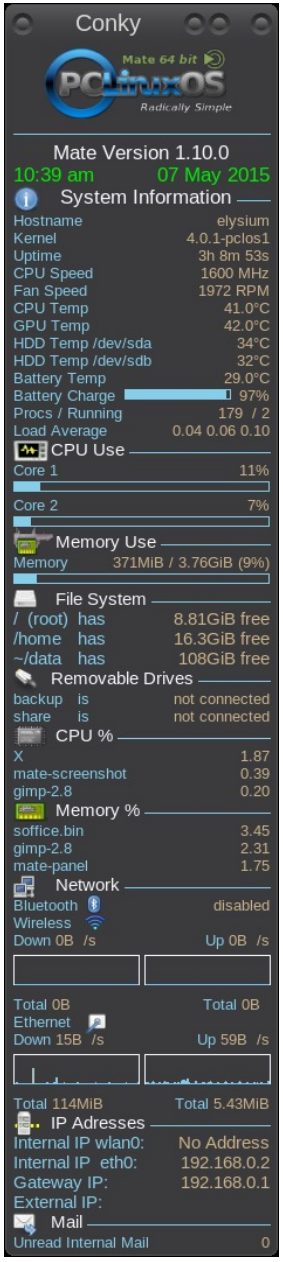
```
4.0.1-pclos1 on

Uptime: 4h 30m 39s
Frequency (in MHz): 2667
Frequency (in GHz): 2.67
RAM Usage: 319MiB/3.76GiB - 8%
Swap Usage: 98.9MiB/8.00GiB - 1%
CPU Usage: 4%
Processes: 178  Running: 0

File systems:
 / 9.75GiB/19.6GiB
Networking:
Up: 0B    - Down: 0B

Name              PID   CPU%   MEM%
 soffice.bin      9190   0.50   3.53
 snmpd            7938   0.50   0.10
 clipit           4477   0.50   0.40
 X                1288   0.50   0.85
```

Néhány alakzat változhat, de ha ezt látod, akkor a Conky működik. Ennek a képernyőnek a



megjelenítéséhez a Conky az alap beállító fájlt használja, ami ez lesz:

```
/etc/conky/conky.conf
```

A fájlt bemásolhatod a saját home könyvtáradba:

```
cp /etc/conky/conky.conf ~/.conkyrc
```

és módosíthatod, vagy akár készíthetsz egy sajátot nulláról, amit én is tenni fogok.

Amikor a conky parancsot kiadod, először egy .conky nevű rejtett fájlt keres a te home könyvtáradban és ha megtalálta, azt használja. Ha nem talál, akkor a saját alap fájlját használja. Ez van, ha -c opció nélkül adod ki a parancsot. Az opció közli Conky-val, hogy egy nevesített beállító fájlt keressen, ami a parancssorban közvetlenül követi, mielőtt az Enter billentyűt lenyomnád. Így lehetőséged van több beállítófájlt használni és Conky példányt futtatni, amik mindegyike mást csinál. Én a konfigurációs fájlomat .conkyrc_mag-nak nevezem el, így az indító parancs a következő lesz:

```
conky -c ~/.conkyrc_mag
```

Hozzászoktam az elején a ~/ használatához, amikor nem a saját könyvtáramban adom ki a parancsot és te is ezt kell, hogy használd, amikor a parancsot automatikusan indító eljárást használsz.

A beállító fájl – 1. rész

Az első szakaszban használatos parancsokat a hivatalos dokumentáció mindet tartalmazza, ami a http://conky.sourceforge.net/config_settings.html címen található. Ugyanakkor, a parancsok és a hozzáfűzött magyarázat zavaros lehet, hacsak nem vagy gyakorlott Conky-szkript író. Néhányat közülük egyáltalán nem fogunk használni. Íme, ahogy a fenti képhez beállítottam a dolgokat.

Úgy tűnhet, mintha sok állítani való lett volna, de a többségük változatlan marad valamennyi konfigurációs fájlodban, kevés változtatással. Vagyis, ha egyszer beállítottad kedvedre, csak apró dolgokat kell változtatni, és néhány sort biztonsággal eltávolíthatsz. Az opciók közül sok parancssorban is beállítható. Írd be, hogy **conky --help** a teljes listáért.

Az opciók közül néhány az átlátszóságért felel, ami csak akkor érvényes, ha olyan ablakkezelőt használsz, ami támogatja a kompozitálást és az engedélyezve van. Szerencsére a PCLinuxOS által használt ablakkezelők rendelkeznek ezzel a tulajdonsággal, de engedélyezni kell, hogy működjék! Mate-nál, amit jelenleg használok, ez a Mate Control Centre → Windows modulnál tehető meg, ahol egy jelölő négyzet van az engedélyezésre és tiltásra. KDE esetén az asztali effektusokat kell engedélyezni.

Íme a ~/.conky_mag beállító része, A számok a sorok előtt nem részei a fájlnak és nem szabad beírni azokat, csak hivatkozásul szerepelnek itt. Az egyes sorokat követő leírás elegendő információt kell adnia ahhoz, hogy más értékeket kipróbálj és lásd a hatását.

Nos, sorról sorra ezek vannak:

1 ***double_buffer yes***

Majdnem biztosan akarod ezt és legyen yes, csökkenti a villogást, amikor a Conky frissíti a képernyőt. Az egyetlen másik elfogadott értékek természetesen a no.

2 ***own_window yes***

(saját ablak) yes, vagy no, ismét legyen yes.

3 ***own_window_argb_visual yes***

a színek meghatározása 1-255 értékkel történik piros, zöld és kék mértékére három színcsatornát tartalmaz – rgb. Ezt az opciót választva egy negyedik, „alpha” csatorna, vagy itt argb jelenik meg. Az alfa csatorna írja le az átlátszóság mértékét. Az értékek 0 teljes átlátszóságtól 255-ig, a teljes átlátszatlanságig terjednek. Ezt általában yes-re kell állítani. Kompozitáló ablakkezelő kell az átlátszósághoz.

4 ***own_window_class Conky***

Eltekinthetünk ettől, mivel a Conky az alap ablakosztály, de a lehetőség hasznosnak bizonyulhat, ha ablakmanipuláló eszközt akarsz használni.

5 ***own_window_hints skip_taskbar***

Ez a sor opciókból karakterláncot vár, amiket vesszők (szóköz nélkül) tagolnak. Az opciók határozzák meg az adott Conky-ablak megjelenését és viselkedését. Az elérhető opciók és hatásaik a következők:

undecorated - eltávolítja az ablakdekorációkat, mint a bezáró gomb és a címsor

below – (alul) a Conky-t a többi ablak alá kényszeríti

above – (felül) engedi, hogy a Conky a többi ablak fölött jelenjen meg

sticky – (ragad) kényszeríti az ablak megjelenítését valamennyi asztalon
skip_taskbar – (eszközsor átugrása) a Conky az eszközsoron nem jelenik meg
skip_pager –(lapozó) a Conky nem jelenik meg a virtuális asztal kiválasztóján

6 own_window_transparent no

Ez álltja be, vagy kapcsolja ki a Conky-ablak átlátszóságát. Csak akkor működik, amikor a kompozitálás engedélyezett.

7 own_window_type normal

(saját ablak típusa) Négy típusú ablakot tud a Conky generálni.
normal – ez az alap
desktop – (asztal) ekkor az ablaktippek többsége engedélyezett
panel – olyan ablakot készít, ami a képernyő széléhez igazodik
override – (felülír) az ablakot nem az ablakkezelő vezérli és előkészületek szükségesek hozzá, hogy használhasd – nem javasolt, hacsak nem tudod pontosan, hogy mit csinálsz.

8 own_window_title Conky

(saját ablakcím) Ez az a cím, ami a címsorban jelenik meg, ha „dekorált” és az átlátszóság letiltva. Alapbeállításként a rendszer hostname jelenik meg.

9 update_interval 5.0

(frissítési intervallum) Megadja képernyőfrissíts gyakoriságát. Az érték másodpercet jelent (5 mp. itt). A túl sűrű frissítés hatására a Conky sokkal több rendszer-erőforrást használ fel. Az értéket tapasztalati úton lehet meghatározni ódosított konfigurációs fájllal. Nagyobb érték általában jobb. Kérdezd meg magadtól „Ténylegesen milyen sűrűn akarom látni az új információkat?”. Az érték függ:

az igényelt adat típusától;
a munkától, amihez az adat szükséges;
a műveletektől és átalakításoktól, amik szükségesek mielőtt megjelenne az adat.

10 alignment top_right

Ahol a Conky ablaka megjelenjen. Az opciók:
top_right (fent jobbra)
top_left (fent balra)
top_middle (fent középen)
bottom_right (lent jobbra)
bottom_left (lent balra)
bottom_middle (lent középen)
Rövidíthetőek igény szerint: tr, tl, tm, br, bl, bm

11 background no

(háttér) A legtöbb felhasználó ezt a sort elhagyhatja. Yes-re az eljárást átküldi a háttérhez a Conky indulásakor.

12 border_width 1

(keretszélesség) A keret lehet háttérszínnel, vagy alapbeállítás a 14. sorban szerinti. Így tölti ki az ablakszél és a tartalom közötti részt. Többnyire egy kis érték megfelel. A keretet akkor húzza meg, ha a 19. sorban yes van.

13 cpu_avg_samples 2

Amikor egy CPU-adat nem pillanatnyi értékének megjelenítése kell, hanem egy bizonyos számú beolvasás átlaga. Minél több beolvasást veszünk, annál pontosabb lesz az érték. Praktikus okokból kis érték megfelelő.

14 gap_x 5

A gap (rés) pixelt jelent a választott igazítási szél (10. sor) és a Conky ablaka között x (balra – jobbra) irányban;

15 gap_y 5

mint az előbbi, csak y (fel – le) irányban.

16 default_color white

(alapszín) Meghatározza az alapszínt, ha más szín nincs megadva és a szegély színe a 19. sorban szerinti. A szín neve is használható itt, ahogy az X11-es színsémában megadott, amit ezen a weblapon szépen megmutatnak: http://www.graphviz.org/doc/info/colors.html
Alternatívaként hexadecimális érték, mint pl. FFFFFF is használható. (Megjegyzés: nincs kezdő # jel.) A GIMP színválasztója hasznos lehet az értékek megállapítására.

17 default_outline_color white

(körvonal alapszín) Az itteni körvonalról a 21. sor intézkedik. A színek a fentiek. szerint.

18 **default_shade_color white**

(árnyék) A szöveg árnyékának színe, ha a 22. sorban engedi. Színek a fentiek.

19 **draw_borders no**

(keret rajzolása) A 12. sorban leírt keret.

20 **draw_graph_borders yes**

Négyszög keretet rajzol a terület köré, ahol megjelenik majd a diagram.

21 **draw_outline no**

Szöveget körvonalasít és keretezi a diagramot azzal a színnel, amit a 17. sorban választottunk.

22 **draw_shades no**

(árnyék rajzolása) Az itt hivatkozott árnyék, a szöveg által vetett.

23 **net_avg_samples 2**

(net mintavétel) A 13. sorban a CPU-val kapcsolatban leírtak szerint működik.

24 **no_buffers yes**

Az itt hivatkozott buffer a memóriáé. A kernel nem szabadítja fel a memóriát azonnal, némi adatot visszatart újrahasznosításra készen, hogy ne töltse újra, de az a memória mindig elérhető felhasználásra, ha kell. Yes-t írva csak azt a memóriaértéket írja ki, ami az adott pillanatban használatban van. No-t írva a teljes, adatokkal lefoglalt értéket írja ki, függetlenül attól, hogy az adat aktív-e, vagy sem.

25 **out_to_console no**

(konzolra írás) Hasznos lehet a yes, ha nem működik rendesen. A Conky által megjeleníteni kívánt kimenet a konzolon is megjelenik, ami megkönnyítheti a hiba helyének, okának feltárását. Normális esetben no-t írjál ide.

26 **out_to_stderr no**

Ez hasonló célú, mint a fenti és további hibakeresési eszköz lehet.

27 **stippled_borders 0**

Ha a keretet engedélyezted, akkor ezáltal tehető szaggatottá, vagy pontozottá. Az érték a megszakítás és a vonal hossza pixelben.
Ha 0 az érték, az folyamatos vonalat jelent. A sor elhagyásának hatása azonos a 0 értékkel.

28 **uppercase no**

(nagybetű) Ha igen, akkor a szöveg nagybetűs lesz.

29 **use_xft yes**

Ez az X freetype betűk támogatását jelenti, ami általában hasznos dolog. Ha nem érted a különbséget, akkor adjál no-t és nézd meg, melyik tetszik neked. Én ezt tettem, majd yes-re állítottam.

30 **xftfont Sans:size=10**

Az alapbeállítás szerinti betűtípus és -méret. A text szekcióban ez felülírható.

31 **own_window_colour 3b3b3e**

(saját ablak színe) Beállítja az ablak háttérszínét, ha az átlátszóság „no”. A színek leírása a 16. sornál.

32 **color1 ghostwhite**

A szövegrész színei előre meghatározhatók color0-tól color9-ig megadva. Ez azért hasznos, mert így könnyű a szkripten belül a kinézetet megváltoztatni és javíthatja a szkript olvashatóságát. Színekre ismét csak a 16. sor a mérvadó.

33 **color2 skyblue** Mint fent

34 **color3 navajowhite3** Mint fent

35 **color4 green2** Mint fent

36 **# End of configuration section** – A beállító rész vége

Ez egy megjegyzés és figyelmen kívül hagyja. Bármi, amit # előz meg megjegyzésként értelmezett és azok akár önálló sorban, akár egy sor végéhez adva az utasítás után, de legalább egy betűközzel elválasztva a parancsot a megjegyzéstől az egyértelmű megkülönböztetés érdekében.

Számos lehetőséggel lehet itt játszani és a dokumentációban még sokkal több van. Ezek azok, amiket én leginkább használok.

A TEXT szekció

A TEXT szó önálló sorban állva jelzi a konfigurációs fájl kimeneti részének kezdetét és a fent leírt konfigurációs rész (szekció) után kell következnie. Nem nagybetűérzékeny, de javasolt a nagybetű használata, hogy könnyebben azonosítható legyen a szekció eleje.

Ami itt van szolgál állandó (statikus) szövegeket megjelenítésére, az előre definiált változók alkalmazására (az értékekre hivatkozó nevek változhatnak), amik a rendszerre vonatkozó információkat adják numerikus, szöveges, vagy grafikus formában, illetve megjelenítik egy zajló folyamat állapotát, esetleg egy zeneszám lejátszásának állapotát. A „Statikus szöveg” alatt azt értem, amikor a kimenet pontosan egyezik az oda begépelttel és nem függ változótól, vagy egy függvény visszaadott értékétől. Nagyjából csinálja egy rendszermonitor, és pontosan ezt teszi a Conky is – kihagytam volna valamit? Ha igen, akkor a Conky bizosan azt is csinálja.

Itt lehetnek megjegyzések, de az üres sort üres sorként jeleníti meg. A szóközök használata kiemelt szereppel bír a text résznél, ezért a megjegyzések írásánál megfontoltan és körültekintően járjunk el.

Minden Conky változót „$” jellel kell bevezetni és ajánlatos a nevét és az argumentumát kapcsos zárójelek közé rakni, ahogy az lent is látható. A zárójelek nem feltétlenül szükségesek, de segítenek a szóköz- és feldolgozási hibák elkerülésében. Zárójelek nélkül a változót követő szöveget szóköznek kell bevezetnie, hogy a Conky pontosan felismerhesse a változót. A szóközt meg fogja jeleníteni, ami nem mindig kívánatos. Szóköz nélkül a Conky nem ismeri fel a változót és az egész karaktersort betűről betűre nyomtatja ki, ha lehetséges. Szokj hozzá, hogy mindig eszerint járj el és akkor kevesebb gondod lesz a kimenetekkel.

Alant a bemutató rendszerfigyelő text szekciója található, a számokat hivatkozásként raktam be, nem részei a szkriptnek.

Megintcsak soronként megyünk, noha a magyarázatok itt egy kicsit hosszabbak is lehetnek.

37 TEXT

A szekció innen indul

38 ***${image /home/felhasználó/Képek/conky/pcloslogo.png -p 27,0 -s 160x60}***

A számot követő teljes szöveg egy sorba legyen. A sor megtörhető fordított törtjellel, ha elég körültekintő vagy, de ha a törtjel nincs a megfelelő helyen az problémákat okozhat. Én az összes képet, amit a Conky-ban akarok megjeleníteni, egy könyvtárban tárolom a home-om Képek könyvtárában. Ez a sor az adott képet az ablakba kirakja. A -p 27,0 pozicionálja a képet 27 pixellel jobbra és 0 pixellel lefelé az ablak bal felső sarkához viszonyítva. A -s 160x60 a képet átméretezi, mégpedig 160 pixel szélesre és 60 pixel magasra.

39 ***${voffset 0}***

A 0 itt semmi mást nem tesz, de egy ettől nagyobb érték utasítja a Conky-t, hogy a kimenetét a számnyi pixellel alacsonyabban kezdje el megjeleníteni. Hasonló módon egy negatív érték a beillesztési pontot felfele tolja. De a beírt sorral sokkal könnyebben tudom a kimenetet irányítani, amennyiben úgy döntenék, hogy változtatok valamit.

40 ***${font Liberation Sans:size=10}***

Ez aparancs időlegesen felülírja a betűtípusra vonatkozó beállításokat, amiket a 30. sor tartalmaz.

41 ***${color2}${hr 1}***

A color2 a 33. sorban megadott, amivel húzunk egy vízszintes, 1px vastag zöld vonalat. A színt 6 karakteres hexadecimális számmal is megadhatjuk, a # jellel, vagy akár anélkül. A sor olyan hosszú lesz, hogy kitöltse a rendelkezésre álló helyet. A lenti kép alját nézd meg, a „Mail”-nél, hogy lásd, hogyan is működik.

42 ***${color1}${alignc}${exec VERSION=`mate-session --version | awk '{ print $2 }'`; echo -n Mate Version $VERSION}***

Ez a sor kezd egy kicsit bonyolultabb lenni. Balról kezdve:
A color1 előre meghatározott színt használva,
a képet az ablak közepéhez igazítja.
Végrehajtja a követő héj-parancsot. A parancs valahogy így értelmezhető: határozd meg az éppen futó mate verzióját, ami az aktuális rendszeremen az 1.10.0-ás.
Add át ezt az értéket az wak-nak, ami levágja a mate-session rész és a maradék 1.10.0-át tárolja a VERSION nevű változóban.
Végül írd ki a „Mate Version” szöveget, amit a VERSION változó tartalma követ, vagyis mi Mate Version 1.10.0-t látunk a képernyőn. Ezzel elérjük, hogy az aktuális verziót mindig mutassa. Más ablakkezelők esetén a parancsot megfelelően módosítani kell. Az exec változó arra való, hogy héj-parancsokat

hajtson végre a kívánt kimenet elérése érdekében. A héj-parancsok némelyike elég erőforrásigényes is lehet, az exec változó a parancsot minden frissítéskor végrehajtja (9. sor) és emiatt a Conky a változónak négy további változatát biztosítja: excep, exei, texeci és execpi.
Az execp változót kényelmi okokból adja, lehetővé teszi más Conky változókkal a shell szkriptedben való keverését és a Conky megfelelően értelmezni fogja (feldolgozza) azokat, de ez elég erőforrásigényes és nem árt, ha a frissítési intervallumot növeled, hogy megfelelően működjön. Az execi változó egy égrehajtási intervallumot ad hozzá, aminek nagyobbnak kell lennie, mint a 9. sorban megadott. Így módodban áll felügyelni, hogy a héj-parancs a 9. sorban meghatározottól függetlenül, mikor hajtódjon végre. A texeci ennél hasznosabb, az execi-hez hasonlóan beiktat egy intervallumot, de ez esetben a shell parancs a saját szálán fut le, az adatokat a Conky-nak átadva és a szálat bezárva a futás után. A 82. sorban látható egy példa a texeci használatára. Végül az execpi egy execp intervallum értékkel kiegészítve. Ahhoz, hogy működjön, a héj-kódnak azelőtt kell lennie, mielőtt az adat visszaküldhető lenne a Conky-nak, vagyis az olyan parancsok mint a tail -f, ami fájlhoz hozzáadást követi a megszakításig, nem fog működni.

43 ***${color4}${time %H:%M %p}${alignr}${time %d %b %Y}***

Most másik előre definiált színt használunk, a zöld egy árnyalatát, hogy megjelenítsük előbb az időt, majd jobbra rendezve a dátumot. Ez zavaró, mert a Conky dátumot és időt eredményező parancsa a time, de formátum karaktersora azonos a Linux date parancsával. A kimenet az én lakóhelyemnek megfelel, de az USA-ban élő felhasználók átállíthatják az utolsó time parancsot {time %b %d %Y} formára. Terminálban kiadva a date –help parancsot megmutatja az idő és dátum formátum elérhető opcióit.

44 ***${image /home/user/Pictures/conky/summary.png -p 0,118 -s 22x22}${color1} System Information ${color2}${hr 1}***

Egy újabb kép méretezve és megfelelően pozicionálva, amit „System Information” sima szöveg követ. A szöveg pozicionálására 8 szóköz van elől és 1 utána, jóllehet a fenti sorban ezek nem látszódnak. Egy jobb pozicionálási eljárásért lásd az 55. sort.
Ezt követően color2 színre váltunk és a fennmaradó rész kitöltésére egy vízszintes vonalat használunk.

45 ***${font Liberation Sans:size=8}${color2}Hostname ${color3}${alignr} ${nodename}***

A betűtípus újbóli átváltása. Ezúttal csak a méretét cseréljük egy kicsit kisebbre.

Váltás color2-re. A fix szöveg „Hostname ” (vedd észre a szóközt utána). Majd csere color3-ra, és a szöveg utána jobbra rendezett. Ez a szöveg a Conky változó értéke ${nodename}, ami az aktuálisan elérhető számítógépnek a host neve, azaz amin éppen dolgozol.

46 ***${color2}Kernel ${color3}${alignr}${kernel}***

Vissza color2-re, a statikus szöveg „Kernel ”, color3 ismét és utána a ${kernel} Conky változó értéke, ami megmutatja az aktuálisan használatban lévő kernel verzióját. Az utolsó érték a Conky ablakában jobbra rendezett.

47 ***${color2}Uptime ${color3}${alignr}${uptime}***

Ez a sor az előbbi formátumát követi, de azt mutatja meg, hogy a rendszer mennyi ideje van bekapcsolva (az utolsó indítás óta).

48 ***${color2}CPU Speed ${color3}${alignr}${freq} MHz***

Ismét egy hasonló kimenet, de ezúttal a CPU jelenlegi (átlag) sebességét mutatja.

49 ***${color2}Fan Speed ${color3}${alignr}${exec sensors | sed -n '/fan1/p' | tr:w -s " " | cut -d" " -f 2 } RPM***

Ezúttal egy kicsit több dolgot kell a shell-ből használnunk a szükséges információ megjelenítéséhez. Color2-vel indítva a „Ventilátor sebesség” kiírása a Conky-ablakba. Váltás color3-ra, a követő szöveget jobbra rendezi, amit a következő parancsból nyerünk: sensors – Megjeleníti a libsensors által szolgáltatott kimentet, ami általában elég sok, ezért vezetjük a sed parancshoz, ami leválasztja a fan1-et tartalmazó sort, amely adat ezután megy a tr parancshoz -s opcióval azért, hogy a ismétlődő szóközöket „kiszedje” (squeeze) a kimenetből. Ezután az tovább megy a cut parancshoz, hogy a kapott az adatot különféle, szóközzel elválasztott csoportokra darabolja fel és a kapott karakterekből a második csoportot, vagy mezőt kiadja. A sor végső része kiír egy statikus szöveget „ RPM”, ami tartalmazza a leválasztó szóközt.

50 ${color2}CPU Temp ${color3}${alignr}${exec sensors | sed -n '/(crit/p' | sed -n '/temp1/p' | cut -d" " -f 9 | sed 's/+//'}

Most a CPU hőmérséklete kell és ez egy újabb komplex parancs.
Azért, hogy a megfelelő kimenetet kapjam, megszabadítva a fölösleges dolgoktól, egészen addig próbálgattam terminálban a parancsot, amíg pontosan azt nem kaptam, amit akartam. A fenti exec parancshoz nyitottam egy terminált és begépeltem a sensors-t. Erre egy teli képernyőt kaptam. Átolvasva alaposan, egy sed paranccsal kerestem meg a kulcsszavakat, amivel a találatok számát szűkítettem.

Ezt követi egy olyan folyamat, ami végrehajtja a kizárásokat és egy közelítő becslést, hogy meghatározza a megfelelő értéket, illetve kiválaszt egyet. A következő sor úgy lett kialakítva, hogy csak a szükséges adatot tartalmazza. Tudom, hogy ez egy fájdalmas út, amin végig kell menni, hogy egy egyszerű információt megkapjunk, de ezt végigjárva párszor könnyebb lesz és sokat tanulsz az olyan hasznos eszközök, mint a sed, awk és cut használatáról.

51 ***${color2}GPU Temp ${color3}${alignr}${exec sensors | sed -n '/temp10/p' | cut -d" " -f 8 | sed 's/+//'}***

Hasonló az 50. sorhoz.

52 ***${color2}HDD Temp /dev/sda ${color3}${alignr}${exec hddtemp /dev/sda | awk '{print $4}'}***

Hasonló az 50. sorhoz.

53 ${color2}HDD Temp /dev/sdb ${color3}${alignr}${exec hddtemp /dev/sdb | awk '{print $4}'}

Hasonló az 50. sorhoz.

54 ${color2}Battery Temp ${color3}${alignr}${exec sensors | sed -n '/temp5/p' | cut -d" " -f 9 | sed 's/+//'}

Hasonló az 50. sorhoz.

55 ***${color2}${if_match "${battery_percent}"=="0"}Battery %${alignr}${color3}Battery not installed${else}Battery Charge${goto 100}${color2}${battery_bar 8,90}${alignr}${color3}${battery_percent}%${endif}***

Ezúttal egy feltételes kifejezést használunk: a kimenetet aszerint akarjuk változtatni, hogy akkumulátor állapot létezik-e. Ha nincs akkumulátor, akkor megállapítjuk a tényt és kilépünk. Ha az akkumulátor jelzi a töltést, akkor ezt telepíteni kell. Megjelenítünk egy rajzolt sávot és a töltöttség százalékát. Először color2-t állítunk. Ezután beolvassuk a ${battery_percent} conky változó értékét és összehasonlítjuk a beolvasott értéket egy abszolút „0” értékkel a ${if_match "${battery_percent}"=="0"} Conky tesztelővel. Egyezés esetén kinyomtatjuk „Battery %”, majd jobbra rendezve kiírja a szöveget „Battery not installed” és továbblép a következő sorra.
Ha az ellenőrzés sikertelen, akkor töltöttséget észlel és akkumulátornak lennie kell. Most a ${else} kifejezés aktiválódik és kiírja „Battery charge”, a beillesztési pontot jobbra 100 pixellel eltolja a ${goto 100}. Ahogy azt a 44. sorban írtuk, a szóközök beillesztése egy módja a szöveg pozicionálásának, de az a hátránya, hogy az függ a betűmérettől, a képek méretétől és más, a sort alkotó elemektől.

A ${goto} conky változót alkalmazva áthelyezi a beillesztési pontot egy rögzített értékre a bal oldali ablakszélhez képest, esetünkben 100 pixellel. Ez a kimenetnek állandó kinézetet nyújt. Váltás color2-re, az akku töltési sávot rajzol 8 pixel magasat és 90 hosszút. Következik a jobbra rendezés és color3, kiíratjuk a ${battery_percent} conky változó értékét, amit egy % jel követ. Ezután jön a feltételes kifejezés lezárása a %{endif} conky változóval.

56 ***${color2}Procs / Running${color3}${alignr}${processes} / ${running_processes}***

Szín váltása color2-re és egy statikus szöveg „Procs / Running” kiírása, váltás color3-ra igazítás változtatása és néhány conky változó értékének kiíratása, amiket „/” választ el.

57 ***${color2}Load Average ${color3}${alignr}$loadavg***

szín, szöveg, szín, igazítás, conky változó értéke (terhelés)

58 ***${font Liberation Sans:size=9}${color1}${image /home/user/Pictures/conky/fft.png -p 0,336 -s 30x20}${goto 46}CPU Use ${color2}${hr 1}***

Ez egy, a 44. sorhoz hasonló fejlécsor, de a beillesztési pont pozicionálásra a ${goto} változót használja.

59 ***${font Liberation Sans:size=8}${color2}Core 1 ${color3}${alignr}${cpu cpu1}%***

Betűméret váltása és color1 használata. A „Core 1” statikus szöveg és simán kiírja. A kimenet színének color3-ra váltása, jobbra rendezés és a kimenet az 1. mag százalékát mutatja. Az egyes magokat cpu1, cpu2 stb. azonosítja, a cpu0 argumentum a teljes cpu-használat kimenetére való. A záró „%” statikus szöveg.

60 ***${color2}${cpubar cpu1 8}***

Megjeleníti az 1. mag cpu-használatát, 8 pixel magas jelzősávval.

61 ***${color2}Core 2 ${color3}${alignr}${cpu cpu2}%***

Mint a 60. sor, csak a 2. magra.

62 ***${color2}${cpubar cpu2 8}***

Megjeleníti az 2. mag cpu-használatát, 8 pixel magas jelzősávval.

```
63 ${font Liberation Sans:size=9}${color1}${image
/home/user/Pictures/conky/ram-use.png -p 0,410 -s
34x26}${goto 46}Memory Use ${color2}${hr 1}
```

Ez az 58. sorhoz hasonló fejléc.

```
64 ${font Liberation Sans:size=8}${color2}Memory
${color3}${alignr}$mem / $memmax ($memperc%)
```

A betű átméretezése ismét, színváltás color2-re, „Memory” fix szöveg beillesztése, váltás color3-ra és jobbra rendezés, a memóriahasználat aktuális értékének beillesztése (erre a 24. sor hatással van), statikus szöveg „\”, a rendszer rendelkezésére álló összmemória, fix szöveg „ ” (szóköz) és a a használatban lévő memória százaléka (a 24. sor erre is hatással van). Ez nem tartalmazza a lapozásra használt részt, arra másik conky változók valók.

```
65 ${color2}${membar 8}
```

Egy 8 pixel magas, a memória használatát jelző sávot jelenít meg color2-vel.

```
66 ${font Liberation Sans:size=9}${image
/home/user/Pictures/conky/hdd.png -p 0,462 -s
20x20}${color1}${goto 46}File System ${color2}${hr 1}
```

Ez az 58. sorhoz hasonló fejléc.

```
67 ${color2}root${goto 60}has${color3}${alignr} ${fs_free /}
free
```

Három belső meghajtó partícióm van, amiket figyelni akarok és ezek a „/” (gyökér), „/home” és „/user/data” alá vannak csatolva. A gyökér és a home partíció mindig csatolva van, így a sávjaik egyszerűen a ${fs_free}conky változót használva jelentenek, ami a partíción szabadon rendelkezésre álló helyet mutatja.

```
68 ${color2}/home${goto 60}has ${color3}${alignr} ${fs_free
/home} free
```

Mint a 67. sor.

```
69 ${color2}${if_mounted /home/user/data}~/data${goto 60}has
${color3}${alignr} ${fs_free /home/user/data}
free${else}~/data${goto 60}is${color3}${alignr}not
mounted${endif}
```

Ez a partíció néha nincs csatolva, hogy védjem a rajta tárolt adatokat, így azzal, hogy tudom az eszközön lévő szabad helyet, a csatolt állapotot is jelzi nekem. Ehhez egy feltételes változót kell használnom. Elsőként ellenőrzöm a ${if_mounted} conky változót. Amennyiben sikeres (igaz) értéket ad vissza, akkor minden a ${else} kifejezésig kimenet lesz és a sor végrehajtása befejeződik. Amennyiben az ellenőrzés sikertelen (hamis) a végrehajtás az ${else} conky változó utáni ágra ugrik és folytatja a szöveg megjelenítését a megfelelő színnel jobbra rendezve.

```
70 ${image /home/user/Pictures/conky/ext-hdd.png -p 0,530 -s
22x22}${color1}${goto 46}Removable Drives ${color2}${hr 1}
```

Ez az 58. sorhoz hasonló fejléc.

```
71 ${font Liberation Sans:size=8}${color2}${if_mounted
/mnt/backup}backup${goto 60}has ${color3}${alignr} ${fs_free /mnt/backup}
free${else}${if_existing /dev/disk/by-label/backup}backup${goto
60}is${color3}${alignr}not mounted${else}backup${goto 60}is${alignr}${color3}not
connected${endif}${endif}
```

Van még két külső meghajtóm is, amiket figyelek. Címkékkel azonosítom a partícióimat, használok még eszközcsomópontokat, mint /dev/sda1 is, vagy az eszköz UUID-jét. A meghajtóimat „backup” és „share” címkékkel láttam el. A meghajtók mindegyike három állapot közül az egyikben van:

Csatolva
Csatlakoztatva, de nem csatolva
Nincs csatlakoztatva

Ez a sor először a ${if_mounted} conky változóval azt ellenőrzi, hogy a meghajtó csatolva van-e. Ha igen, akkor jelenti azt és a sort bezárja. Ha a meghajtó nincs csatolva, akkor a sor végrehajtása az ${else} conky változó utáni kódrészre ugrik és megpróbálja a meghajtót megkeresni az ${if_existing} conky változó segítségével.

A Linux az általa ismert meghajtókra és partíciókra vonatkozóan a /dev rendszerkönyvtárban hivatkozást szerepeltet. A három hely, amiket ellenőrizni kell:

/dev/disk/by-label
/dev/disk/by path (device node)
/dev/disk/by-uuid.

Ha az ellenőrzés – ${if_existing /dev/disk/by-label/backup} partíciót talál, akkor a második esettel állunk szemben – a „connected but not mounted” (csatlakozva, de nem csatolva) és ezt jelenti, a végrehajtás befejeződik. Ha a teszt nem találja a meghajtót, akkor egyértelmű, hogy arra a harmadik eset illik – „Not connected” (nincs csatlakoztatva). Jelenti és a sor végrehajtása véget ér.

72 `${color2}${if_mounted /mnt/share}share${goto 60}has ${color3}${alignr} ${fs_free /mnt/share} free${else}${if_existing /dev/disk/by-label/share}share${goto 60}is${color3}${alignr}not mounted${else}share${goto 60}is${alignr}${color3}not connected${endif}${endif}`

Mint a 71. sor

73 `${font Liberation Sans:size=9}${image /home/user/Pictures/conky/processor.png -p 0,580 -s 30x20}${color1}${goto 46}CPU % ${color2}${hr 1}`

Fejléc, akár az 58. sor

74 `${font Liberation Sans:size=8}${color2}${top name 1} ${color3}${alignr}${top cpu 1}`

A ${top} conky változó a folyamatokról jelent a CPU-használat függvényében a legnagyobbtól a legkisebbig rangsorolva. A változó 2 argumentumot használ, típus és besorolás. Itt a típust és az 1.-nek rangsoroltat jelentjük (a legnagyobb CPU-használat).

A típus lehet az egyik ezek közül: név, pid, mem, mem_res, mem_vsize time, uid, user, io_perc, io_read, vagy io_write.

A top tízet lehet jelenteni.

75 `${color2}${top name 2} ${color3}${alignr}${top cpu 2}`

Mint a 74. sor, de a 2. besorolású.

76 `${color2}${top name 3} ${color3}${alignr}${top cpu 3}`

Mint a 74. sor, de a 3. besorolású.

77 `${font Liberation Sans:size=9}${image /home/user/Pictures/conky/memory.png -p 0,644 -s 30x20}${color1}${goto 46}Memory % ${color2}${hr 1}`

Az 58. sorhoz hasonlóan fejléc.

78 `${font Liberation Sans:size=8}${color2}${top_mem name 1}${color3}${alignr}${top_mem mem 1}`

Mint a 74. sor, de az 1. besorolású, memóriahasználat szerint rangsorolva a CPU-használat helyett.

79 `${color2}${top_mem name 2}${color3}${alignr}${top_mem mem 2}`

Mint a 74. sor, de a 2. besorolású, memóriahasználat szerint rangsorolva a CPU-használat helyett.

80 `${color2}${top_mem name 3}${color3}${alignr}${top_mem mem 3}`

Minta 74. sor, de a 3. besorolású, memóriahasználat szerint rangsorolva a CPU-használat helyett.

81 `${font Liberation Sans:size=9}${image /home/user/Pictures/conky/network.png -p 0,709 -s 20x20}${color1}${goto 46}Network ${color2}${hr 1}`

Az 58. sorhoz hasonlóan fejléc.

82 `${font Liberation Sans:size=8}${color2}Bluetooth${image /home/user/Pictures/conky/bluetooth.png -p 60,724 -s 18x18}${if_match "${texeci 10 cat /proc/acpi/ibm/bluetooth | grep status | awk '{print $2}'}"=="enabled"}${alignr}${color3}enabled${else}${alignr} ${color3}disabled${endif}`

A laptopom egy Thinkpad és a bluetooth státusza a /proc/acpi/ibm/bluetooth nevű fájlban van tárolva. Más gépek esetén is kell legyen hasonló bejegyzés. A bluetooth státusza lehet „enabled” (engedélyezett), vagy „disabled” (nem engedélyezett). Ez a sor a ${if_match} conky változót használja az aktuális állapot meghatározására és jelenti. A keresés ${texeci} conky változót alkalmaz, ami új folyamatot indít a kód végrehajtására és ismétli azt 10 másodperces időközönként. Ez lehetővé teszi, hogy a többi, a Conky további részei tegyék a dolgukat, miközben ez alacsony prioritással fut. A sor kódjának többi része mostanra már ismerős kell legyen.

83 `${if_up wlan0}${color2}Wireless${image /home/user/Pictures/conky/network-wireless.png -p 61,740 -s 18x18}${alignr}Signal Strength ${color3}${wireless_link_qual}%${else}${color2}Wireless${image /home/user/Pictures/conky/network-wireless.png -p 61,740 -s 18x18}${endif}`

Ellenőrzi a wifi adaptert a wlan0-t, hogy csatlakozik-e és ha igen, akkor jelenti a jelerősséget. Az ellenőrzés lehet sokkal szigorúbb a konfigurációs résznél erre beírt résszel. A ${if_up_strictness} conky változó egy argumentumot elfogad, ami lehet:

up – (él) de nincs kapcsolat
link – él és van kapcsolat

address – él, kapcsolódva és IP-címmel rendelkezik.

Ha az adapter nincs csatlakozva, akkor csak a kép és a „Wireless” szó jelennek meg.

84 ***${color2}Down ${color3}${downspeed wlan0}/s ${alignr}${color2}Up ${color3}${upspeed wlan0}/s***

Továbbra is a vezeték nélküli adapterről, ez jelenti a feltöltési és letöltési sebességet.

85 ***${color1}${downspeedgraph wlan0 25,107 7ac6f7 d7ba8e -l -t} ${alignr}${color1}${upspeedgraph wlan0 25,107 7ac6f7 d7ba8e -l -t}***

Vezeték nélküli adapter letöltési sebességét egy 25 pixel magas és 107 pixel széles sávval jelzi ki, két különböző színt használva, amiket a # nélküli hexadecimális számok határoznak meg. Az -l argumentum logaritmikus skálát határoz meg, hogy a kis értékek is látszódjanak, a -t argumentum pedig lehetővé teszi, hogy a szín a mért adat függvényében hőmérséklet változás szerűen módosuljon.

86 ***${color2}Total ${color3}${totaldown wlan0} ${color2}${alignr}Total ${color3}${totalup wlan0}***

A vezeték nélküli adapter teljes letöltése és feltöltése.

87 ***${color2}Ethernet${image /home/user/Pictures/conky/ethernet.png -p 60,826 -s 20x20}***

Az eth0 Ethernet adapter kimenetének kezdete.

88 ***${color2}Down ${color3}${downspeed eth0}/s ${alignr}${color2}Up ${color3}${upspeed eth0}/s***

Mint a 84. sor, csak az eth0-ra.

89 ***${color1}${downspeedgraph eth0 25,107 7ac6f7 d7ba8e -l -t} ${alignr}${color1}${upspeedgraph eth0 25,107 7ac6f7 d7ba8e -l -t}***

Mint a 85. sor, csak az eth0-ra.

90 ***${color2}Total ${color3}${totaldown eth0} ${color2}${alignr}Total ${color3}${totalup eth0}***

Mint a 86. sor, csak eth0-ra.

91 ***${font Liberation Sans:size=9}${image /home/user/Pictures/conky/ip-addresses.png -p 0,911 -s 22x22}${color1}${goto 46}IP Adresses ${color2}${hr 1}***

Az 58. sorhoz hasonlóan fejléc.

92 ***${color2}Internal IP wlan0: ${alignr}${color3}${addr wlan0}***

A wlan0 IP-címe.

93 ***${color2}Internal IP eth0: ${alignr}${color3}${addr eth0}***

Az eth0 IP-címe.

94 ***${color2}Gateway IP:${alignr}${color3} ${gw_ip}***

Az átjáró IP-címe.

95 ***${color2}External IP: ${alignr}${color3}${texeci 3600 wget http://checkip.dyndns.org -O - -o /dev/null | cut -d : -f 2 | cut -d \< -f 1}***

Külső weblapról leszedett IP-cím, egyórás időközönként. Biztonsági okokból az érték nem jeleníthető meg nyíltan, ezért a fenti képről eltávolítottam.

96 ***${font Liberation Sans:size=9}${image /home/user/Pictures/conky/mail.png -p 0,998 -s 20x20}${color1}${goto 46}Mail ${color2}${hr 1}***

Fejléc, mint az 58. sor.

97 ${font Liberation Sans:size=8}${color2}Unread Internal Mail ${alignr}${color3} ${new_mails /var/spool/mail/user}

Új belső e-mail ellenőrzése. Használatra itt kell beállítani ezt a tulajdonságot. Én postfix-et használok, így a szolgáltatásokról, mint a cron, vagy biztonsági másolatot készítő szkript, tudom fogadni az értesítéseke.

Conky használata

A Conky kipróbálásához készíts konfigurációs fájlt a home könyvtáradba és add hozzá a fenti beállító részeket, beleértve az 1-től 36. sort is. Ezt a TEXT szó kövesse, önálló sorban. Mentsd a fájlt valami olyan névvel, mint conky-sablon. A

sablon apróbb módosításokkal a legtöbb Conky-beállításra, amiket csak létre akarsz hozni, használható lesz

Ezután a „TEXT” részből a 38-tól a 97. sorig adj hozzá egy önálló sort. Teljesen lényegtelen, hogy melyiket (ezek csak példák), de bármelyik sor, ami képet jelenít meg, úgy kell beállítani, hogy egy tényleges képre mutasson, csakúgy mint a partíciók és hálózati interfészek esetén. Válassz egy egyszerű sort az indításhoz és ne próbálj túl sok sort hozzáadni még, különben problémák lesznek, ha valami nem működne rendesen. Mentsd újra a fájlt valami megfelelő néven pl. ~/.conkyrc_mag.

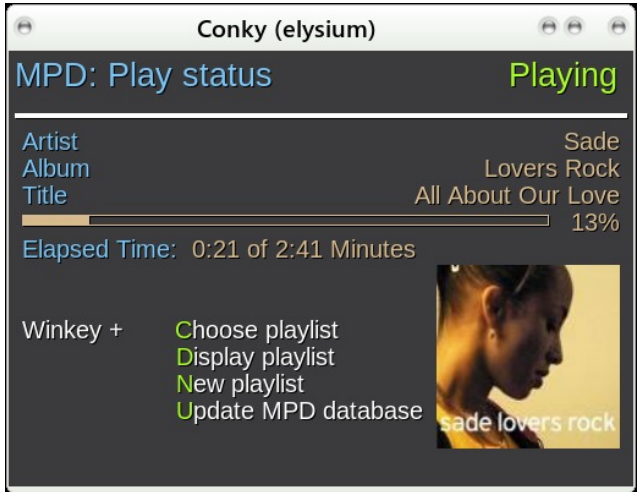
Nyiss egy terminált és hívd meg a létrehozott fájlt conky -c ~/.conkyrc_mag formában (vagy ahogy a fájlodat elnevezted és ne felejtsd el a pontot az elejéről).

Ha minden jó, akkor látnod kell annak a sornak a kimenetét. Ha nem, akkor ellenőrizd a beírást. Ha javítottál és mentetted a fájlt és készítettél sablon fájlt is, akkor azon is csináld meg a javításokat. Csináld meg most, mielőtt elfelejtenéd. Ha már fut, akkor hozzátehetsz és elvehetsz sorokat a TEXT szekcióban, illetve változtathatsz, hogy elkészítsd a saját, személyre szabott Cony-dat. Próbálj megváltoztatni néhány értéket a beállító szekcióban, hogy lásd a hatásukat. Használj másik betűt, vagy betűket. Biztosan meg kell változtatni a színeket, hogy a te témádhoz és hangulatodhoz jobban menjen. Ez egyéni dolog, és sosem leszel elégedett a munkáddal, amíg az neked tetsző nem lesz.

Ha eddig eljutottál, akkor nagyjából tiszában vagy már a Conky beállításával. A mintául szolgáló szkript nekem megfelel és az alapok nagy részét lefedi. Itt van néhány példa, hogy ötletet nyerhess. Most már eleget tudsz ahhoz, hogy képes legyél valami hasonlót készíteni te is.

Az Interneten számos ehhez hasonló példát találhatsz, közülük pár ennél sokkal fejlettebbet is. Néhány annyira fejlett, hogy a Conky-nak támogatás kell megjelenítésükhöz és hamarosan azokkal is foglalkozni fogunk. Előbb nézzünk meg néhány további dolgot, amire az alap Conky képes.

A Conky nem nem korlátozódik kizárólag rendszer-információk megjelenítésére. Elég sokoldalú eszköz lehet. Például a Magazin számára a Music Player Daemon-ról (MPD) írott egyik cikkemben bemutattam, hogy a Conky-val miképpen tudja megjeleníteni a játszott szám címét, állapotát és az album borítóját, A beállító fájl a cikk része, vagyis nem ismétlem meg itt, de a Conky valahogy így nézett ki.



RSS

RSS információk Conky-val megjelenítéséhez nyiss egy sablon fájlt és mentsd **~/.conkyrc_rss**-ként.

A konfigurációs szekcióba írd:

change ***own_window_title conky to own_window_title Breaking News***

Add hozzá ***minimum_size 400 250 és maximum_width 400***

Ezzel egy fix méretű kijelződ lesz még akkor is, amikor az ablak tartalma megváltozna.

A sablon fájlodat egyelőre üres TEXT szekcióval mentetted, ezért a TEXT szekcióhoz add hozzá ezeket a sorokat:

```
${color2}${rss http://feeds.bbci.co.uk/news/world/rss.xml
30 feed_title}
```

```
${color1}${rss    http://feeds.bbci.co.uk/news/world/rss.xml
30 item_titles 10 4}
```

Csak ennyire kell egy új Conky készítéséhet, ha már van alap sablonod. Mielőtt megmutatnám az eredményt, elmagyarázom, hogy az egyes sorok mire valók

Az **${rss}** conky változó a következő argumentumokat fogadja el:

uri, interval, action, num_par és space_in_front. Az utóbbi kettő opcionális..

Az **url** a rss-adás címe, esetünkben http protokolt használ, de ez a conky változó bármilyen protokollal képes működni, amit a curl parancs kezelni képes.

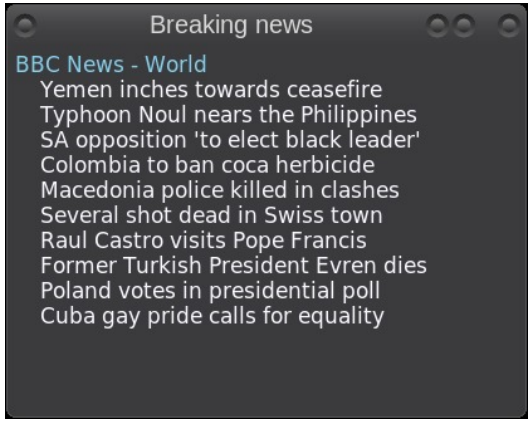
Az **interval** a frissítések közötti idő percekben.

Az **action** azt adja meg, hogy a információ mely részét akarod megjeleníteni. Ez lehet:

feed_title	az rss-adás címét;
item_title num_par	bármely x számú (num_part) elemet, a számozás 0-tól van
item_desc num_par	x számú elem szövegét;
item_titles	0-tól num_part-ig számú elem címét. Ezzel a művelettel az opcionális space_in_front (szóközök előtte) argumentum az elemeket a szóközök számával egyező mértékben tolja be.

num_par a rss-feed elemszáma.

Ez az rss Conky változó saját szálon fut, és a Conky-t a konfigurációs fájl módosítása után újra kell indítani, hogy lásd a változásokat.



Az item_descr argumentummal szolgáltatott sorok szövege elég hosszú is lehet és a Conky nem túl jól tördeli a sorokat. Az itt megjelenített információ arra elég, hogy jelezzen nekem mindent, amit ami esetleg további vizsgálatot igényel.

Jóllehet ezt a tulajdonságot önálló alkalmazásként mutattam be, az rss-tartalom része lehet egy sokkal összetettebb Conky képernyőnek.

Átlátszóság

A Conky képernyőjét könnyen átlátszóvá tehetjük a konfigurációs részben megváltoztatva két sort. Ennek működéséhez a kompozitálást, vagy az asztali effektusokat engedélyezni kell.

Változtasd a következő sort

```
own_window_transparent no
```

erre

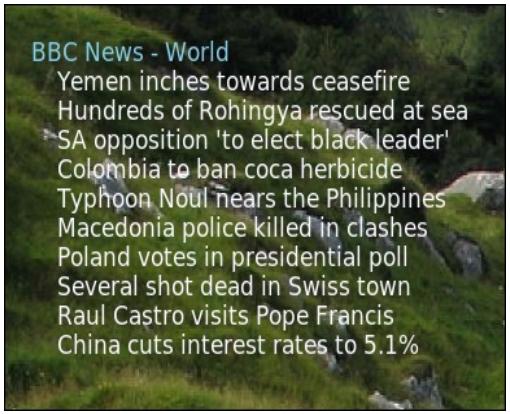
```
own_window_transparent yes
```

és az

```
own_window_hints skip_taskbar
```

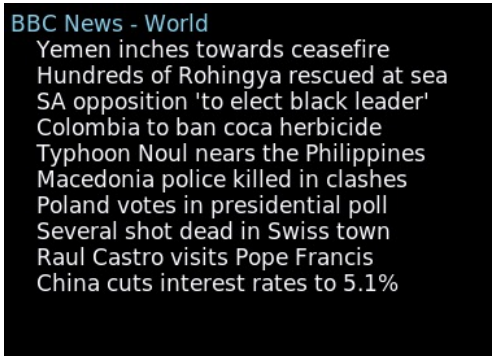
sort erre

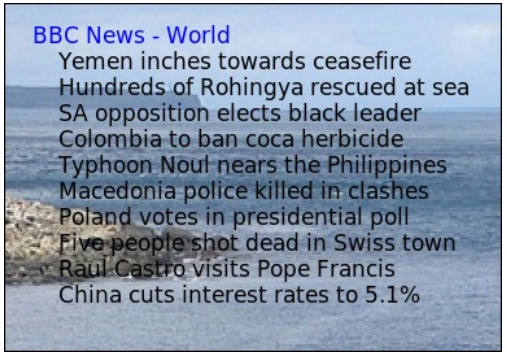
```
own_window_hints skip_taskbar,undecorated
```



Jól mutatja az átlátszóság egyik problémáját. A fenti két kép ugyanazt a Conky képernyőt mutatja ugyanazon háttér két eltérő részén. A színeket természetesen lehet módosítani, hogy ezeket kezelni tudja, de ez olyasmi, amit figyelembe kell venni, amikor Conky-képernyőt tervezünk.

A következő két kép ugyanazt a dolgot mutatja be úgy, hogy előbb a kompozitálás ki van kapcsolva, illetve amikor be van kapcsolva, de a szövegszínt a jobb kontraszt érdekében háttérképhez igazítottuk. A harmadik képen látható, amikor a kompozitálás engedélyezett!





Sablonok

Sablonok használata egyszerűsítheti a konfigurációs fájlt, különösen akkor, amikor hasonló típusú információt kell megjeleníteni több sorban.

A .conkyrc_mag a 67. sorában ezt tartalmazza:

```
${color2}root${goto 60}has${color3}${alignr} ${fs_free /} free
```

A 68. sor majdnem teljesen megegyező.

Ha egy ilyen sablon definíciót adok a konfigurációs szekcióba:

```
template1 ${color2}\1${goto 60}has$alignr}${color3}${fs_free \2}
```

akkor a 67. és a 68. sor ilyen lesz:

${template1 root /}
${template1 home /home}

A 69. sor ez volt:

```
${color2}${if_mounted /home/user/data}~/data${goto 60}has ${color3}${alignr} ${fs_free /home/user/data} free${else}~/data${goto 60}is${color3}${alignr}not mounted${endif}
```

Ilyen formára egyszerűsödik:

```
${if_mounted /home/pete/data}${template1 ~/data /home/pete/data}${else}~/data${goto 60}is${color3}${alignr}not mounted${endif}
```

Ezzel nagymértékben egyszerűsíthető a TEXT szekció fő része, ami sokkal egyszerűbbé teszi a dolgok kezelését.

A sablonok így működnek.

Egy sablon definíciója, vagy leírása a beállító, vagy forrás fájl konfigurációs részében történik. Igen, tudom, a terminológia kicsit zavaró, de ne foglalkozz vele, hamarosan rendben lesz. Egy sablon olyan névvel definiálható, ami tartalmazza a „template” szót és egy 0-tól 9-ig terjedő számot. A sablonunk neve „template1”. A sablon-definíció további része a fájl TEXT szekciójának soraival azonos módon épül fel. A kivétel az, hogy ott, ahol a TEXT szekció sorában beírnánk az elem nevét, amiről szeretnénk, hogy a Conky jelentsen, a sablonban „helyőr” szerepel, ami egy fordított törtjellel bevezetett szám. Azt hiszem, a számozás 1-től 9-ig terjedhet, de ebben nem vagyok biztos – használd 1 és 9 között, egyébként 1-4-nél többre sosem lesz szükséged! Azoknak, akik tudják, mi az, a helyőrök hasonlóak a kiterjesztett szabályos kifejezésekben használtakkal.

Amikor a fájl TEXT szekcióban egy sablon nevet talál, akkor azt lecseréli a sablon-definíció teljes tartalmára néhány apróbb módosítással. A változtatás annyit tesz, hogy a helyőrzőket lecseréli a sablon nevét közvetlenül követő értékre

Példánkban a \1-et lecseréli az sablon meghívót azonnal követő elemre, a \2 esetében pedig a második elemre stb. Így ha a \1-et root-ra cseréli és a \2-őt /-re, akkor a

```
${template1 root /}
```

erre változik

```
${color2}root${goto 60}has$alignr}${color3}${fs_free /}
```

és a

```
${template1 home /home}
```

pedig erre

```
${color2}home${goto 60}has$alignr}${color3}${fs_free /home}
```

A helyőrzők arra az elemre cserélődnek le, amik a sablon meghívásánál közvetlenül követik. A sorban a többi elemet pl. ${goto 60} szokásos módon értelmezi.

Ha ez kicsit zavarosnak tűnne, számodra először az volt, akkor próbálj ki néhány példát és biztosan megérted a koncepció lényegét.

Mérők és sávok

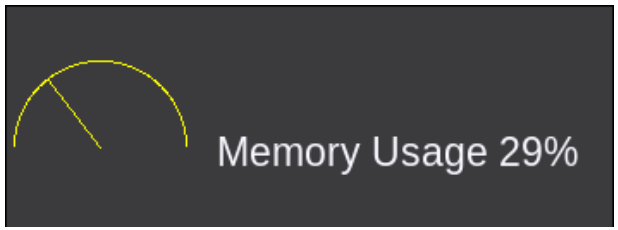
Jelzősávokat már láttunk: akkumulátor feszültség és CPU jelzősáv. Folyamatos képi visszajelzést adnak a megjelenített elem használtságáról. Az akkumulátor jelzősávja mutatja, hogy az akku mennyire van feltöltve, de nem számmal, hanem a sáv maximális hosszához viszonyított hosszal jelzi azt. A magassága és az ablakon belüli elhelyezése változtatható, de semmi egyéb. Hasznos, de nem túl érdekes.

A Conky rendelkezik egy másik arányos megjelenítővel, amit még nem néztünk meg – elliptikus mérő, ami ugyanúgy dolgozik, mint a nyomásmérő, vagy a sebességmérő óra. Az órák mindarra használhatók, amiket a sávok be tudnak mutatni: akku töltöttsége, memóriahasználat ...

Ennek a két sornak az alkalmazása

```
${font Liberation Sans:size=14}
${color yellow}${memgauge 50,100}${color1} Memory Usage ${memperc}%
```

ezt a kimenetet produkálja:



Oké, működik és láthatod, hogy mit akartam mondani. Bizonyos helyzetekben ez is elég, nem kell több. De hé! Ez Linux és mi nem nyugszunk. Mi ÜTŐSET akarunk.

Ha ÜTŐSET akarsz, akkor csinálunk ÜTŐSET! De ehhez dolgoznod kell. Lua és Cairo kell. Ezekkel a második részben foglalkozunk.