Conky

Conky est une application qui délivre en temps réel des informations sur votre système.
Ces informations s'affichent sur votre bureau, dans une fenêtre classique, dans une barre de status comme dzen ou dans une barre de statut propre à un gestionnaire de fenêtres, comme dwm ou wmfs.
Conky est livré avec de nombreuses options internes mais peut aussi faire appel à des scripts (bash,python,lua...),
ce qui lui permet d'afficher à peu près tout ce que vous voulez et comme vous le voulez.

La configuration de Conky passe par l'édition du fichier **.conkyrc** situé généralement dans votre dossier utilisateur.
Ce fichier se divise en deux parties:

- les **"fonctions"** déterminent l'emplacement, la dimension, le type de fenêtre, la couleur du texte...
- les **"variables"** déterminent la nature des informations à afficher.

La séparation se fait par la ligne "TEXT" dans le conkyrc; tout ce qui vient après TEXT sera affiché dans votre conky.

Sommaire

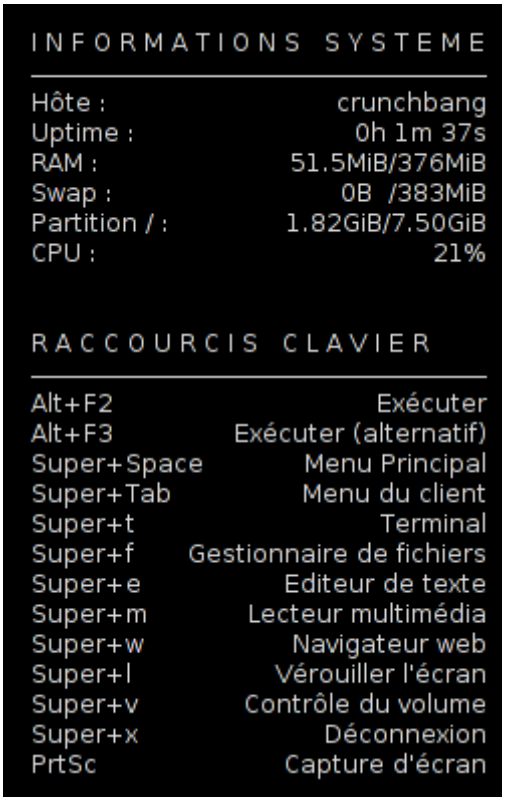
Conky....1
Configuration....2
Conky classique....2
Conky en ligne....4
Conky fenêtré....6
Conky dans une barre de statut....9
Conky dans dzen2....10
Conky en pipe-menu....12
Conky diaporama....13
Les Scripts....17
Les Scripts Lua....17
Conky Weather....26
Conclusion et Recommandations....26
Liens....26
Conky : les fonctions....27
Conky : les variables....32

Configuration

Le net est rempli de tutoriels plus ou moins complets à propos de conky. Une des meilleures sources d'information (en dehors du site officiel) reste Conky Pitstop, où vous pourrez trouver de nombreuses astuces en français (merci wlourf). Vous trouverez une liste détaillée complète des fonctions et variables employées par conky dans ce wiki, mais quelques exemples seront plus explicatifs: en passant par ces configurations, vous aurez un apperçu assez complet des possibilités de conky.

Conky classique

Prenons pour exemple le conky de départ de crunchbanglinux, une des seules distributions proposant un conky par défaut. (le fichier est placé dans votre /home/$USER/.conkyrc).



conky crunchbanglinux

Nous allons détailler le conkyrc qui permet cet affichage.

```
##############################################
#  Settings
##############################################
background yes                ## permet à conky de tourner en arrière-plan
use_xft yes                   ## utilise la police xft
xftfont sans:size=9           ## nom de la police xft à utiliser
xftalpha 1                    ## opacité de la police xft, ici 1=opaque
update_interval 1.0           ## conky se recharge toutes les secondes
total_run_times 0             ## conky se relance à l'infini
own_window yes                ## conky utilise sa propre fenêtre
own_window_transparent yes    ## conky dessine une fenêtre transparente
```

```
own_window_type desktop      ## type de fenêtre de conky: bureau
own_window_hints undecorated,below,sticky,skip_taskbar,skip_pager  ## propriétés de
la fenêtre
double_buffer yes            ## empêche le clignotement
minimum_size 200 200         ## taille minimale largeur200,hauteur200
maximum_width 240            ## largeur maximale 240 px
draw_shades no               ## n'affiche pas les ombres sous le texte
draw_outline no              ## n'affiche pas le contour de la police
draw_borders no              ## n'affiche pas les bordures
draw_graph_borders no        ## n'affiche pas le contour des graphiques
default_color d8d8d8         ## couleur des textes et bordures
default_shade_color 000000   ## couleur des ombres
default_outline_color d9d7d6 ## couleur du contour de la police
alignment top_right          ## alignement de conky, ici en haut à droite
gap_x 12                     ## décallage horizontal, ici 12px
gap_y 24                     ## décallage vertical, ici 24px
no_buffers yes               ## 'buffers' non pris en compte dans le calcul de
mémoire.
uppercase no                 ## police en mode normal (Maj/min)
cpu_avg_samples 2            ## nombre de processeurs pris en compte: ici 2
override_utf8_locale yes     ## pour avoir les caractères accentués
###############################################
#  Output
###############################################
TEXT                                    ## ici commencent les informations à
afficher
I N F O R M A T I O N S   S Y S T E M E     ## texte brut sans variables (pas de $)
${hr}                                   ## ligne horizontale
Hôte :$alignr$nodename                       ##--|
Uptime :$alignr$uptime                       ##  |
RAM :$alignr$mem/$memmax                     ##  |--variables notez que les {} ne
sont pas
Swap :$alignr$swap/$swapmax                  ##  |  nécessaires pour les
variables sans arguments
Partition / :$alignr${fs_used /}/${fs_size /}    ##  |
CPU :$alignr${cpu cpu0}%                     ##--|
                                          ## lignes vides: notez que conky prend
en compte les sauts de lignes.
R A C C O U R C I S   C L A V I E R          ## texte brut sans variables (pas de $)
${hr}                                     ## ligne horizontale
Alt+F2$alignr Exécuter                    ## ici commencent la description des
raccourcis clavier #!
Alt+F3$alignr Exécuter (alternatif)
Super+Space$alignr Menu Principal
Super+Tab$alignr Menu du client
Super+t$alignr Terminal
Super+f$alignr Gestionnaire de fichiers     ## notez la variable "$alignr"
(alignement à droite)
Super+e$alignr Editeur de texte             ## qui permet une meilleure présentation
Super+m$alignr Lecteur multimédia
Super+w$alignr Navigateur web
Super+l$alignr Vérouiller l'écran
Super+v$alignr Contrôle du volume
Super+x$alignr Déconnexion
PrtSc$alignr Capture d'écran
```

Hormis la description des raccourcis claviers, ce conky ressemble beaucoup à la configuration de conky par défaut (situé dans /etc/conky/conky.conf). Cette configuration de base est suffisamment explicite pour qui veut un conky simple.

Conky en ligne

Le conky en ligne, nommé ainsi car il tient sur une seule ligne (ou deux), est une façon élégante d'afficher ses informations système. Cest aussi très pratique, car combiné à la configuration des “marges” d'openbox, vos informations resteront toujours visibles. La configuration de ce genre de conky est un peu plus complexe car elle fait entrer en jeu les variables d'alignements et de décallage ainsi que quelques scripts.

conky en ligne

Examinons le conkyrc correspondant:

```
background yes
use_xft yes                       ## une police à chasse fixe est recommandée pour
xftfont Terminus:pixelsize=10     ## ce genre de conky.
xftalpha 0.8
update_interval 1.0
total_run_times 0
own_window yes
own_window_transparent no         ## la fenêtre est opaque pour être toujours visible
own_window_colour 161616
own_window_type override          ## type "override": conky 100% indépendant
own_window_title Infos System
own_window_hints undecorated,below,sticky
double_buffer yes
minimum_size 1024 0               ## largeur mini=maxi=taille de l'écran, conky
maximum_width 1024                ## ressemble à une barre de tâche
draw_shades no
draw_outline no
draw_borders no
border_inner_margin 0
draw_graph_borders no
default_color grey60                    ## couleur par défaut
color1 FF3A3A # rouge                   ##--|
color2 FFC13A # orange                  ##  |
color3 68FF3A # vert                    ##  |--couleurs additionnelles
color4 3AFFFD # bleu ciel               ##  |
color5 443AFF # bleu marine             ##--|
alignment tr
gap_x 0
gap_y 0
no_buffers yes
uppercase no
text_buffer_size 1024
top_name_width 10             ## limite la taille des noms des processus dans '$top'
cpu_avg_samples 2
short_units yes
override_utf8_locale yes
use_spacer none
if_up_strictness address      ## attend d'obtenir une IP avant de déclarer le réseau
ouvert
###############################################
#  Output
###############################################
TEXT
  Pkg ${texeci 28800 ~/bin/debupdates.sh} update(s)${goto 150}BAT ${battery_bar 8,40
```

```
BAT0} $battery${goto 376}hdd:${ibm_temps 2}°C  cpu:${ibm_temps 0}°C  gpu:${ibm_temps
3}°C${goto 576}vol ${if_match "$ibm_volume" == "mute"}mute${else}${ibm_volume}/14$
{endif} - pcm ${mixer PCM}%${goto 760}R/W ${diskiograph 10,40} $diskio/s${goto
920}HDD ${fs_bar 8,40 /} ${fs_used_perc /}%
${if_up eth0} ${color4}DL ${downspeedgraph 8,40 eth0} ${downspeed eth0}/s${color3}$
{goto 156}UP ${upspeedgraph 8,40 eth0} ${upspeed eth0}/s${color}${goto 296}${color
grey80}gMail ${texeci 120 python ~/bin/gmail.py}${else}  no connection${endif}${goto
376}${color1}CPU ${cpugraph 8,40} $cpu% > ${top name 1}${goto 576}${color2}RAM $
{membar 8,40} $memperc% > ${top_mem name 1}${color}${goto 760}${color #CCCCCC}LOAD
$loadavg${color}${alignr 10}${time %a %d %b   %I:%M}
```

Voici donc à quoi ressemble un conkyrc en ligne. Passons en revue les éléments remarquables de cette configuration:

Notez l'emploi quasi systématique de la variable '$goto' qui permet de placer un élément à un endroit précis. Cette variable est très utile car elle permet d'éviter aux autres de 'bouger': en effet, une variable placée par exemple derrière '$cpu' sans '$goto' sera continuellement en mouvement en fonction de la valeur de '$cpu' (1%,10%,100%).

- La première expression remarquable est aussi la première sur la ligne: elle nous renvoie le nombre de paquets pouvant être mis à jour: '$texeci' execute le script “debupdates.sh” toutes les 28800 secondes(8 heures). Le script:

```
#!/bin/bash
# conky script for displaying available updates
# in Debian. This script assumes you are in the
# sudo group and require no password for root
# access. Add something as such to your conkyrc:
#${color}APT: ${color D7D3C5}${execi 28800 ~/bin/debupdates.sh}

sudo apt-get -qy update > /dev/null
NUMOFUPDATES=$(sudo aptitude search "~U" | wc -l)
echo $NUMOFUPDATES
```

Vous comprenez maintenant que '$texeci' éxécute ce sript, puis en lit le résultat afin de l'afficher à l'endroit voulu.

- la seconde exression remarquable est le 'if_match' du volume ibm:

```
vol ${if_match "$ibm_volume" == "mute"}mute${else}${ibm_volume}/14${endif}
```

pourquoi demander à conky de m'afficher 'mute' si 'mute' ?? et bien, pour pouvoir obtenir ces deux affichages: soit “vol mute” soit “vol 5/14” remarquez que ”/14” n'apparait pas dans le premier exemple.

- la seconde ligne commence par '${if_up eth0}, en effet, il semble inutile de demander à conky de délivrer des informations réseau si vous n'êtes pas connecté (j'aurais du y penser pour les mises à jour Debian ;)):

```
${if_up eth0} ${color4}DL ${downspeedgraph 8,40 eth0} ${downspeed eth0}/s${color3}$
{goto 156}UP ${upspeedgraph 8,40 eth0} ${upspeed eth0}/s${color}${goto 296}${color
grey80}gMail ${texeci 120 python ~/bin/gmail.py}${else}  pas de connexion${endif}
```

Dans cet exemple vous comprenez que si(if) eth0 est actif, conky affichera les graphiques (up/downspeedgraph), la quantité de données transférées (up/downspeed) ainsi que le nombre de nouveaux mails, sinon(else) “pas de connexion”, fin des conditions(endif).

Nous retrouvons notre variable '$texeci' mais associé à “python” pour obtenir le nombre de nouveaux gmails. Le script:

```
#! usr/bin/env python3
# simple mail script by tinara

import imaplib
import re

# If you don't want to use the SSL version for IMAP
# Mailbox = imaplib.IMAP4('host','port')
# gmail ex: host = imap.gmail.com, port = 993
Mailbox = imaplib.IMAP4_SSL('host', 'port')
rc,resp = Mailbox.login('username','password')

if rc == 'OK':
     rc, message = Mailbox.status('INBOX', "(UNSEEN)")
     unreadCount = re.search("UNSEEN (\d+)",str( message[0])).group(1)
     print(unreadCount)
else :
     print('Fail to connect')

Mailbox.logout()
```

Comme dans l'exemple du script debian, '$texeci' va éxécuter puis lire le résultat du script gmail.py.

- La dernière expression remarquable est située dans la dernière section:

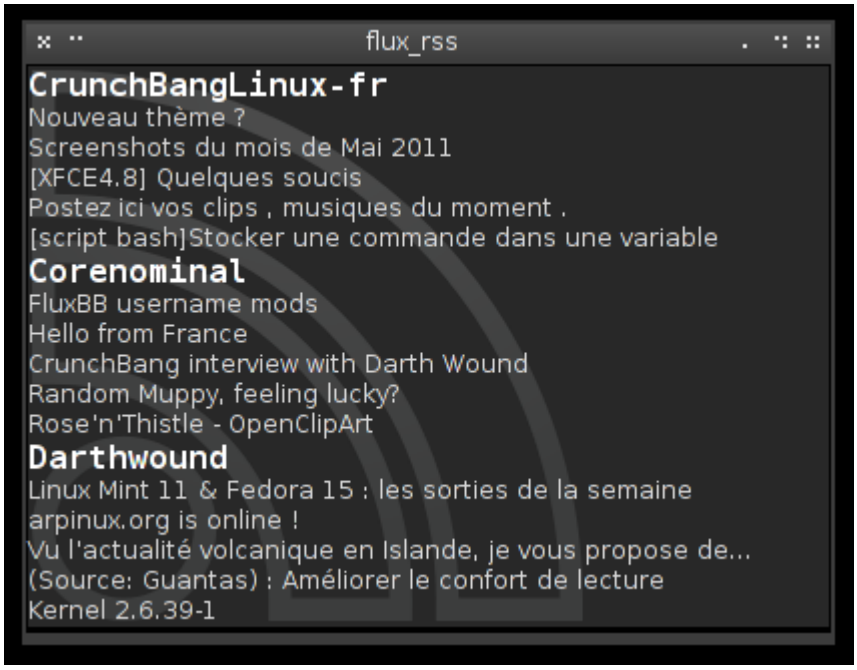
```
${alignr 10}${time %a %d %b   %I:%M}
```

Notez ici l'emploi de '${alignr 10}' qui aligne l'heure sur la droite du conky avec une marge de 10 pixels. Notez aussi que la variable '$time' respecte le formatage des arguments: les espaces sont restitués dans l'affichage de l'heure.

Vous savez maintenant comment positionner votre conky, lui donner les propriétés appropriées, afficher vos informations à l'emplacement désiré et faire appel à des scripts pour receuillir des informations supplémentaires.

Conky fenêtré

Dans certains cas, conky peut être affiché de façon provisoire; pour donner des infos météo, des flux rss, ou pour pouvoir le déplacer et surtout, le fermer quand on veut. Dans cet exemple, nous utiliserons les flux rss:



conkyrss

Comme vous pouvez le constater, cette fenêtre est normale :) .

Analysons le conkyrc afin de voir comment inclure des flux rss, afficher conky dans une fenêtre classique, afficher une image.

```
##############################################
#  Settings
##############################################
background yes
use_xft yes
xftfont sans:size=9
xftalpha 1
update_interval 1.0
total_run_times 0
own_window yes
own_window_transparent no                     ## fenêtre opaque
own_window_type normal                        ## fenêtre de type normal afin de
pouvoir être déplacée/fermée.
own_window_title flux_rss          ## indique un titre précis pour conky
own_window_hints sticky            ## conky apparaitra sur tous les bureaux virtuels
double_buffer yes
minimum_size 400 280               ## on fixe la taille de la fenêtre:
maximum_width 400                  ## à tester selon les flux choisis.
draw_shades no
draw_outline no
draw_borders no
draw_graph_borders no
default_color d8d8d8
color1 white
default_shade_color 000000
default_outline_color d9d7d6
alignment top_middle               ## on centre la fenêtre sur l'écran à l'ouverture
gap_x 0
gap_y 150
border_inner_margin 0      ## pas de marge entre le texte et la bordure
border_outer_margin 0      ## pas de marge entre la bordure et le bord de la fenêtre.
no_buffers yes
```

```
uppercase no
cpu_avg_samples 2
override_utf8_locale yes
##############################################
#  Output
##############################################
TEXT
${image ~/.conky/conkyrssbg.png -p 0,0}${if_up eth0}${color1}${font
Monospace:size=12:bold}CrunchBangLinux-fr${font}${color}
${rss http://crunchbanglinux-fr.org/forum/extern.php?action=feed&type=rss 10
item_titles 5}
${color1}${font Monospace:size=12:bold}Corenominal${font}${color}
${rss http://corenominal.org/feed/ 60 item_titles 5}
${color1}${font Monospace:size=12:bold}Darthwound${font}${color}
${rss http://darthwound.tumblr.com/rss 60 item_titles 5}${else} pas de connexion
réseau.${endif}
```

- La première variable affiche l'image de fond du conky (avec l'emblême rss). Notez que cette variable n'est pas seule sur une ligne: si c'était le cas, conky rajouterais une ligne vide à l'affichage. Pour rajouter une image de fond, vous pouvez soit prendre n'importe quelle image et la redimensionner avec conky (option -s LxH), soit préparer une image aux dimensions de votre conky, ce que j'ai fait avec cette image:

conkyrss background

Il vous reste juste à déterminer la position par rapport au coin supérieur gauche (-p 0,0) et le tour est joué.

- Vous notez le retour de notre '$if_up eth0' qui empêche conky d'aller chercher des informations si le réseau n'est pas disponible. Vous pouvez également constater que les variables et options se combinent : $color1 + $font pour afficher le titre dans une couleur et une police différente, le retour aux valeurs par défaut se faisant par un '${font}${color}' en fin de ligne.
- Passons aux flux rss: il existe une variable pour afficher les titres des flux rss (ex : “${rss http://darthwound.tumblr.com/rss 10 feed_title}”) mais j'ai préféré éditer les noms moi-même. J'utilise ici l'option “item_titles 'n'” qui permet d'afficher les 'n' premiers titres du flux,

mais vous pouvez toujours appeler les titres de flux un par un avec l'option “item_title 'n'” où 'n' représente le numéro du titre (NB: les titres commencent à 0).

- Le petit plus: vous avez certainement noté la différence de fréquence d'appel des flux rss selon les sites: le premier flux concerne les derniers posts actif du forum crunchbanglinux-fr, réglé pour être rafraichit toutes les 10 minutes: “${rss http://crunchbanglinux-fr.org/forum/extern.php?action=feed&type=rss **10** item_titles 5}”, tandis que les autres flux pointant eux sur des blogs, se renouvellent toutes les heures: “${rss http://darthwound.tumblr.com/rss **60** item_titles 5}”.
- Le gros plus: ce style de conky se lance grâce à l'argument “-c” qui indique à conky qu'il ne doit pas utiliser son conkyrc par défaut (/home/$USER/.conkyrc) mais un autre fichier de configuration; il se lance comme ceci:

```
conky -c ~/.conky/conkyrc-rss
```

il est donc très facile de rajouter cette ligne de commande à votre menu openbox ou créer un raccourcis clavier correspondant.

La section dans votre menu pour conkyrss (~/.config/openbox/menu.xml):

```
<item label="flux rss">
    <action name="Execute">
        <execute>
            conky -c ~/.conky/conkyrc-rss
        </execute>
    </action>
</item>
```

La section pour un raccourcis clavier (~/.config/openbox/rc.xml):

```
<keybind key="W-A-r">
  <action name="Execute">
    <startupnotify>
      <enabled>true</enabled>
      <name>Conky RSS</name>
    </startupnotify>
    <command>conky -c ~/.conky/conkyrc-rss</command>
  </action>
</keybind>
```

*pour plus d'informations sur la manière d'éditer votre menu (menu.xml)ou votre fichier de configuration openbox (rc.xml), veuillez vous rendre sur les pages dédiées: menu.xml - rc.xml.

Conky dans une barre de statut

C'est le conky qui s'affiche dans une barre de statut. Dans ce cas, conky ne s'affiche pas réellement: il s'éxécute et est “lu” par la statusbar. L'exemple classique est l'envoi de conky dans la barre de status de dwm.

Pour cela , vous aurez besoin d'un conkyrc particulier car il indiquera à conky de ne rien afficher !! ... En fait, les informations seront envoyé vers STDOUT et récupérés par dwm. Voici comment procéder:

le conkyrc:

```
##############################################
#  Settings
##############################################
background no
out_to_console yes
out_to_x no
update_interval 1.0
total_run_times 0
uppercase no
short_units yes
use_spacer none
if_up_strictness address
##############################################
#  Output
##############################################
TEXT
[info-system]-[cpu:${cpu}%]-[ram:${memperc}%]-[i/o:${diskio}/s]-[hdd:$
{fs_used_perc /}%]-[up:${upspeedf eth0}/s]-[down:${downspeedf eth0}/s}]--[${time %d/
%m - %I:%M}]
```

Vous constatez que les fonctions sont réduites lorsque conky ne s'affiche pas dans X. Bien sûr, les informatons étant supposées s'afficher dans une barre de statut, conky doit tenir sur une ligne.
Il se lance depuis le ~/.xinitrc, juste avant le lancement de la session dwm:

```
## set statusbar ##
conky | while true; read line; do xsetroot -name "$line"; done &

## launch WM ##
exec ck-launch-session /usr/local/bin/dwm
```

Conky dans dzen2

dzen2 est une barre d'information hautement configurable qui va lire les informations délivrées par conky comme pour la barre de statut de dwm. l'avantage de dzen2 est qu'il accepte les formats de couleurs, polices, peut afficher des images (xbm,xpm), graphiques ou des barres de progressions, et qu'il peut s'afficher partout sur votre écran. pour plus d'informations sur l'installation, la configuration et les ooptions de dzen2, je vous laisse visiter sa page dédiée sur ce wiki. pour un affichage simple de conky, une installation standard suffit, pour les débianistes:

```
# apt-get install dzen2
```

le fichier de configuration du conky présenté dans le screenshot ressemble à celui utilisé pour dwm mais ajoute les options pour dzen:

```
out_to_x no
out_to_console yes
update_interval 1.0
total_run_times 0
use_spacer none

TEXT
 ^i(/home/arp/.arp_setups/dzicons/cal.xbm) ^fg(\#ccc)${time %d/%m}^fg()
^i(/home/arp/.arp_setups/dzicons/clock.xbm) ${time %I:%M}\
  ^fg(\#ff4500)^i(/home/arp/.arp_setups/dzicons/cpu.xbm) ^fg(\#ccc)${cpu}%\
```

```
 ^fg(\#ee2c2c)^i(/home/arp/.arp_setups/dzicons/temp.xbm) ^fg(\#ccc)${ibm_temps 0}°C\
 ^fg(\#87ceeb)^i(/home/arp/.arp_setups/dzicons/monitor.xbm) ^fg(\#ccc)${loadavg}\
  ^fg(\#00bfff)^i(/home/arp/.arp_setups/dzicons/mem.xbm) ^fg(\#ccc)$memperc% > $mem\
  ^fg(\#ffd700)^i(/home/arp/.arp_setups/dzicons/home.xbm) ^fg(\#ccc)${fs_used_perc /}
%\
${if_mounted /media/arp500} ^fg(\#000)^i(/home/arp/.arp_setups/dzicons/usb.xbm)
^fg(\#ccc)${fs_used_perc /media/arp500/}%${endif}${if_mounted /media/lacie300}
^fg(\#1e90ff)^i(/home/arp/.arp_setups/dzicons/usb.xbm) ^fg(\#ccc)${fs_used_perc
/media/lacie300/}%${endif}\
 ^fg()${if_match ${battery_percent BAT0}
>=26}^fg(\#7cfc00)^i(/home/arp/.arp_setups/dzicons/bat_full_01.xbm)${endif}\
${if_match ${battery_percent BAT0} <=25}
^fg(\#CC0000)^i(/home/arp/.arp_setups/dzicons/bat_low_01.xbm)^fg()${endif} ^fg(\#ccc)
${battery_percent}%\
 ^fg()${if_up eth0}  ^i(/home/arp/.arp_setups/dzicons/net_wired.xbm)\
 ^fg(\#00cd00)^i(/home/arp/.arp_setups/dzicons/net_up_02.xbm) ^fg(\#ccc)${upspeedf
eth0}\
 ^fg(\#ffa500)^i(/home/arp/.arp_setups/dzicons/net_down_02.xbm) ^fg(\#ccc)$
{downspeedf eth0}\
^fg() ${endif}\
${if_match "$ibm_volume" == "mute" }^fg(\#000000)
^i(/home/arp/.arp_setups/dzicons/spkr_04.xbm)^fg(\#ccc) mute ${else}^fg(\#FFDF1D)
^i(/home/arp/.arp_setups/dzicons/spkr_01.xbm)^fg(\#ccc) ${ibm_volume}/14${endif}
```

note l'emploi du “\” en fin de ligne permet l'annulation du saut de ligne dans conky mais aide à une meilleure lisiblilité du fichier.

comme vous pouvez le constater, les options passées à conky sont “encadrées” par les options passées à dzen2:

- ^fg(): pour la couleur du texte/de l'image
- ^i() : pour l'adresse de l'image à afficher

les images doivent être au format *xbm (facilement éditables avec the Gimp). une archive contenant 67 icones est disponible ici.

ce fichier conky est appelé par dzen2 grâce à un script (que vous prendrez soin de rejouter à votre fichier ~autostart selon votre configuration):

```
#!/bin/sh
RC="$HOME/.conkyrc_dzen"
FG="white"
BG="#404240"
ALIGN="left"
WIDTH="744"
HEIGHT="12"
FONT="-*-terminus-medium-*-*-*-12-*-*-*-*-*-*-*"
XPOS="30"
YPOS="756"

conky -d -c $RC | dzen2 -fg $FG -bg $BG -ta $ALIGN -w $WIDTH -h $HEIGHT -x $XPOS -y
$YPOS -fn $FONT -dock &
exit 0
```

sources du script: minull conky par xeNULL.

Conky en pipe-menu

Il est possible d'afficher des informations venant de conky dans le menu openbox en utilisant les pipe-menus. *Cette fonction n'est pas accessible pour les variables '$cpu','$down/upspeed*'*

Nous allons utiliser pour cela l'exemple précédent: reprenons le conkyrc-dwm, effacons les variables et renommons-le ~/.conkyrc-ob (modifiez selons vos préférences mais pensez à faire correspondre le nom dans la suite de l'exemple):

```
##############################################
#  Settings
##############################################
background no
out_to_console yes
update_interval 1.0
total_run_times 0
uppercase no
short_units yes
use_spacer none
if_up_strictness address
##############################################
#  Output
##############################################
TEXT


```

Il faut impérativement laisser une ligne vide après “TEXT”.

Ce fichier est vide ... à quoi sert-il ?? Il sert de fichier de configuration pour les commandes passées dans ce script:

```
#!/bin/sh
LOAD=$(conky -c ~/.conkyrc-ob -q -i 1 -t '${loadavg}')
TEMP=$(conky -c ~/.conkyrc-ob -q -i 1 -t '${acpitemp}°C')
echo "<openbox_pipe_menu>"
echo "<item label=\"load:$LOAD\" />"
echo "<item label=\"cpu temp:$TEMP\" />"
echo "</openbox_pipe_menu>"
```

Sauvegardez-le comme ~/bin/sys_info.sh (modifiez selons vos préférences mais pensez à faire correspondre le nom dans la suite de l'exemple). Et rendez-le éxécutable:

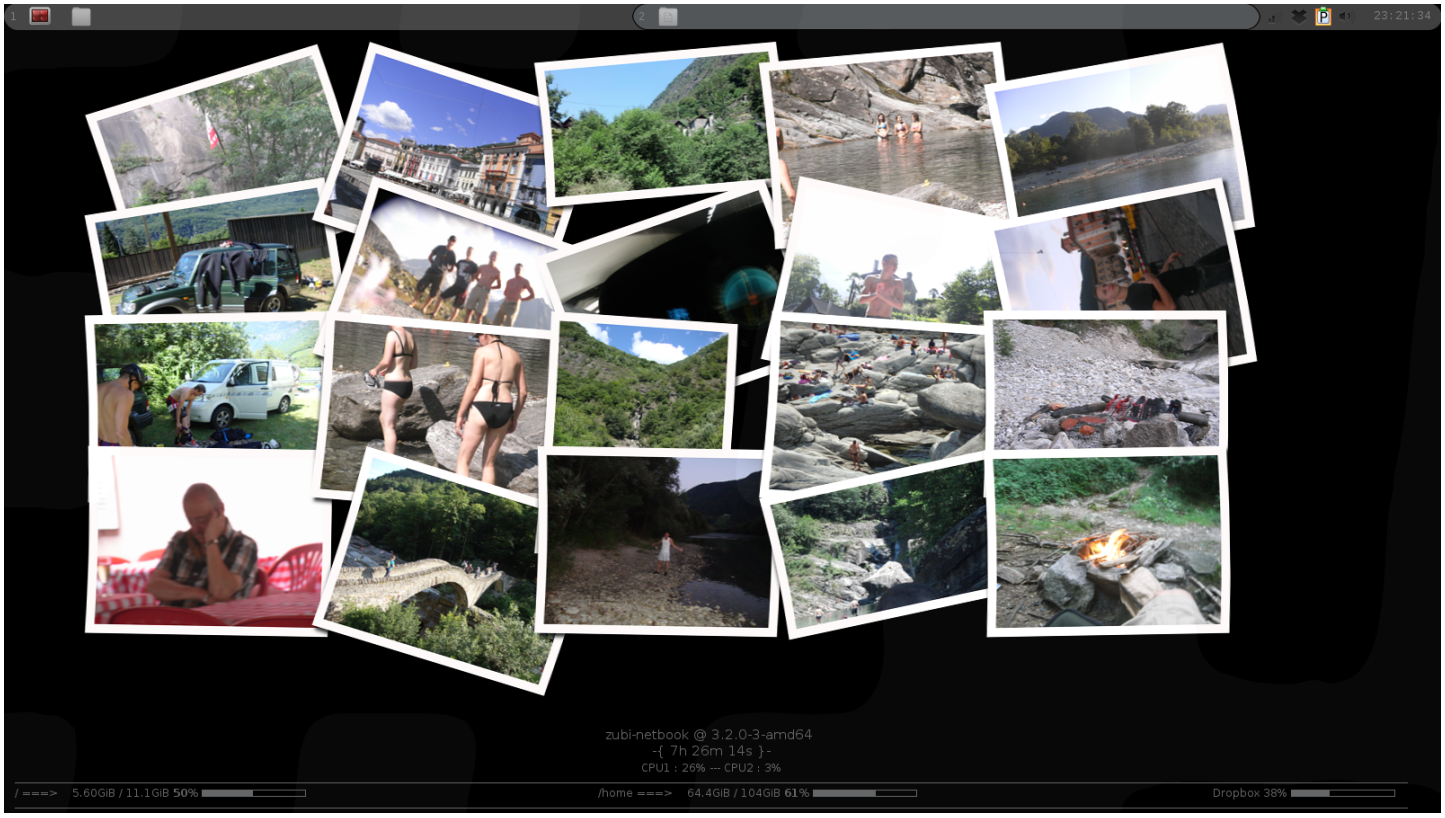
```
chmod +x ~/bin/sys_info.sh
```

Il ne vous reste plus qu'à appeler ces informations à travers un pipe-menu. Insérez cette ligne à l'endroit où vous désirez voir le menu apparaitre:

```
<menu id="infos system" label="Infos Système" execute="~/.sys_info.sh" />
```

Si vous désirez changer les informations affichées, editez simplement votre “~/bin/sys_info.sh”.

Conky diaporama



sources et image: zubi33 sur le forum crunchbang-fr.

il est possible de faire afficher des images par conky grâce à la variable '${image}'. couplé à un script qui choisit des images de façon aléatoire, vous obtenez un diaporama de vos photos ou images favorites:

l'exemple qui suit affiche 20 images sur un écran de 1600×900. pour une configuration différente, il faudra éditer le script et le conkyrc.
pour obtenir un diaporama sur votre bureau avec conky, il faudra tout d'abord choisir vos images, les placer dans un dossier, puis les modifier afin d'ajouter une bordure et de redimensionner vos photos. enfin, vous aurez besoin d'un script pour choisir vos images de façon aléatoire et les afficher dans conky.

- création du dossier contenant les images à afficher en diaporama:

```
$ mkdir ~/images/galerie_conky
```

- copie des photos/images dans le dossier de destination
- création du script de modification des images (dépend de **imagemagick**):

```
$ touch ~/images/galerie_conky/polaroid.sh
```

- édition du script de modification des images avec votre éditeur de texte préféré:

```
#!/bin/bash

# Script permettant de générer des images avec cadre et rotation dans le
dossier ./final
# en utilisant le remplacement.
# ce script s'utilise dans le répertoire contenant les images en jpg.
```

```
#Script start

echo

for fichier in *.jpg
do
  ls -l "$fichier"  # Liste tous les fichiers de $PWD (répertoire courant).

done

echo; echo

for fichier in *.jpg
do

ANGLE=$(($RANDOM % 20)) # Angle aléatoire
if [ $(($RANDOM % 2)) -eq 1 ]
then
    ANGLE="-"$ANGLE
fi
convert "$fichier" -gravity center -resize 240x240 -bordercolor snow -background
black -polaroid $ANGLE "$fichier.png"


echo "Conversion de \"$fichier\"".
done

echo

for fichier1 in *.JPG

do
  ls -l "$fichier1"  # Liste tous les fichiers de $PWD (répertoire courant).

done

echo; echo

for fichier1 in *.JPG
do

ANGLE=$(($RANDOM % 20)) # Angle aléatoire
if [ $(($RANDOM % 2)) -eq 1 ]
then
    ANGLE="-"$ANGLE
fi
convert "$fichier1" -gravity center -resize 240x240 -bordercolor snow -background
black -polaroid $ANGLE "$fichier1.png"


echo "Conversion de \"$fichier1\"".
done

echo

for fichier1 in *.jpeg

do
  ls -l "$fichier1"  # Liste tous les fichiers de $PWD (répertoire courant).
```

```
done

echo; echo

for fichier1 in *.jpeg
do

ANGLE=$(($RANDOM % 20)) # Angle aléatoire
if [ $(($RANDOM % 2)) -eq 1 ]
then
    ANGLE="-"$ANGLE
fi
convert "$fichier1" -gravity center -resize 240x240 -bordercolor snow -background
black -polaroid $ANGLE "$fichier1.png"


echo "Conversion de \"$fichier1\"".
done

echo
# création du répertoire de destination
mkdir final/
# nettoyage des images originales
rm *.jpg
rm *.JPG
rm *.jpeg
# déplacement des images modifiées dans le dossier de destination
mv *.png final/

echo

exit 0
```

ce script ajoute une bordure à toutes les images jpg présentes dans le dossier où il se trouve, les redimensionne en 240×240 pixels maximum sans déformation de l'image, les sauvegarde au format png dans un dossier ~/images/galerie_conky/final, puis éfface les originaux.

- rendre le script exécutable:

```
$ chmod +x ~/images/galerie_conky/polaroid.sh
```

- création du fichier de configuration de conky:

```
$ touch ~/images/galerie_conky/conkyrc
```

- édition du conkyrc avec votre éditeur de texte préféré:

```
background yes
update_interval 30
total_run_times 0
own_window yes
own_window_transparent yes
own_window_type override
own_window_hints undecorated,below,sticky,skip_taskbar,skip_pagers
double_buffer yes
minimum_size 1500 3000
maximum_width 1500
alignment tl
gap_x 10
gap_y 10
imlib_cache_size 0
text_buffer_size 2048
```

```
TEXT
${execpi 300 ~/images/galerie_conky/galleryconky.sh ~/images/galerie_conky/final}
```

- création du script de choix d'images aléatoires:

```
$ touch ~/images/galerie_conky/galleryconky.sh
```

- édition du script de choix d'images

```
#!/bin/bash

# Photo aléatoire pour conky

FILES=( /home/$USER/images/galerie_conky/final/*.png )
PICTURE1="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE2="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE3="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE4="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE5="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE6="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE7="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE8="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE9="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE10="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE11="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE12="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE13="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE14="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE15="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE16="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE17="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE18="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE19="${FILES[$(($RANDOM % ${#FILES[@]}))]}"
PICTURE20="${FILES[$(($RANDOM % ${#FILES[@]}))]}"

#echo $PICTURE

## Affichage dans conky
echo "\${image $PICTURE1 -p 0,0}"
echo "\${image $PICTURE2 -p 0,150}"
echo "\${image $PICTURE3 -p 0,300}"
echo "\${image $PICTURE4 -p 0,450}"
echo "\${image $PICTURE5 -p 250,0}"
echo "\${image $PICTURE6 -p 250,150}"
echo "\${image $PICTURE7 -p 250,300}"
echo "\${image $PICTURE8 -p 250,450}"
echo "\${image $PICTURE9 -p 500,0}"
echo "\${image $PICTURE10 -p 500,150}"
echo "\${image $PICTURE11 -p 500,300}"
echo "\${image $PICTURE12 -p 500,450}"
echo "\${image $PICTURE13 -p 750,0}"
echo "\${image $PICTURE14 -p 750,150}"
echo "\${image $PICTURE15 -p 750,300}"
echo "\${image $PICTURE16 -p 750,450}"
echo "\${image $PICTURE17 -p 1000,0}"
echo "\${image $PICTURE18 -p 1000,150}"
echo "\${image $PICTURE19 -p 1000,300}"
echo "\${image $PICTURE20 -p 1000,450}"


exit 0
```

c'est dans ce script que vous pouvez régler le nombre et la position des images sur votre bureau.

- rendre le script galleryconky exécutable:

```
$ chmod +x ~/images/galerie_conky/gallery_conky.sh
```

il ne vous reste plus maintenant qu'à choisir vos images, les placer dans le dossier ~/images/galerie_conky, puis d'exécuter le script polaroid.sh afin de modifier vos images, et enfin de lancer votre diaporama avec la commande:

```
$ conky -c ~/images/galerie_conky/conkyrc
```

Les Scripts

la meilleure source d'info en français sur les scripts utilisable par conky reste Conky Pitstop-fr maintenu par wlourf!!

Les Scripts Lua

Conky gère le lua depuis sa version 1.7.2 et c'est avec joie que les divers dev et passionnés de conky se sont lancés à la conquète de nouvelles configurations.
Conky ne pouvait afficher jusque là que du texte ou des barres de progression rudimentaires. Avec l'arrivée du lua , les moyens de distiller une information sont devenus quasi infinis.
Londonali1010 est la première à avoir publié un conky s'appuyant sur un script LUA, il me semblait donc légitime de lui réserver mon premier exemple. Je la remercie encore pour son fabuleux travail. Petit exemple de simplicité et d'élégance par Londonali1010:

ring-meters by londonali1010

Ce résultat est obtenu par le chargement d'un script LUA par conky. Le script original est intégralement commenté par Londonali mais .. en anglais... Pour l'exemple, je vous propose ici une version traduite raccourcie :

```
--Ring Meters by londonali1010 (2009)
--traduction par arpinux <http://arpinux.org> (2011)
--Ce script dessine les valeurs en pourcentage sous forme de cercles.
--Il est complètement personnalisable; toutes les options sont décrites dans le
script.

--IMPORTANT: si vous utilisez la fonction 'cpu, elle causera une erreur de
segmentation
-- si elle essaye de dessiner un cercle tout de suite. La condition 'if' de la ligne
255
-- utilise un delai pour être certain que cela n'arrive pas. Elle calcule le delai
par
-- rapport au nombre de rafraichissement de conky depuis son lancement.
-- Generalement, une valeur de 5 est suffisante, donc, si vous relancez conky toutes
-- les 5 secondes, utilisez update_num>5 comme condition (par défaut).
-- Si vous relancez conky toutes les 2 secondes, vous devriez changer cette valeur
-- pour update_num>3; à l'inverse, si vous relancez conky toutes les 0,5 secondes,
-- vous devriez utilisez update_num>10. AUSSI, si vous modifiez votre conkyrc,
-- il est préférable de faire "killall conky; conky" pour le recharger, autrement,
-- le update_num ne sera pas rechargé et il y aura une erreur.

-- Pour appeler ce script dans conky, ajouter ces lignes avant TEXT dans votre
conkyrc
-- (si il est placé dans ~/scripts/ring.lua):
--    lua_load ~/scripts/rings-v1.2-fr.lua
--    lua_draw_hook_pre ring_stats

--Changelog:
-- + v1.2 -- Added option for the ending angle of the rings (07.10.2009)
-- + v1.1 -- Added options for the starting angle of the rings,
--           and added the "max" variable, to allow for variables that output
--           a numerical value rather than a percentage (29.09.2009)
-- + v1.0 -- Original release (28.09.2009)

settings_table = {
     {
           -- Editer cette table pour personnaliser vos cercles.
           -- Vous pouvez créer plus de cercles en ajoutant plus d'éléments à cette
table.
           -- "name" est le type d'info à afficher; vous pouvez utiliser 'cpu',
'memperc', 'fs_used_perc', 'battery_used_perc'...
           name='time',
           -- "arg" est l'argument passé à "name"; si vous utiliseriez ${cpu cpu0},
'cpu0' est l'argument,
        --  si vous n'utiliseriez pas d'arguments, employer ''.
           arg='%I.%M',
           -- "max" est la valeur maximum du cercle. Si conky affiche le résultat en
pourcentage, employer 100.
           max=12,
           -- "bg_colour" est la couleur de fond du cercle.
           bg_colour=0xffffff,
           -- "bg_alpha" est la valeur alpha du fond (0=transparent, 1=opaque).
           bg_alpha=0.1,
           -- "fg_colour" est la couleur de premier plan du cercle.
```

```
            fg_colour=0xffffff,
            -- "fg_alpha" est la valeur alpha du premier du cercle (0=transparent,
1=opaque).
            fg_alpha=0.2,
            -- "x" and "y" sont les coordonnées du centre du cercle en partant du
coin supérieur gauche du conky.
            x=165, y=170,
            -- "radius" est le rayon du cercle.
            radius=50,
            -- "thickness" est l'épaisseur du cercle, répartit autour du rayon.
            thickness=5,
            -- "start_angle" est l'angle de départ du cercle, en degrés, compté
depuis le sommet
         --  dans le sens des aiguilles d'une montre. peut être positif ou négatif.
            start_angle=0,
            -- "end_angle" est l'angle de fin du cercle, en degrés, compté depuis le
sommet
         -- dans le sens des aiguilles d'une montre. peut être positif ou négatif,
         -- mais doit être supérieur à l'angle de départ.
            end_angle=360
      },
      {
            name='time',
            arg='%M.%S',
            max=60,
            bg_colour=0xffffff,
            bg_alpha=0.1,
            fg_colour=0xffffff,
            fg_alpha=0.4,
            x=165, y=170,
            radius=56,
            thickness=5,
            start_angle=0,
            end_angle=360
      },
(...
.
...)
      {
            name='fs_used_perc',
            arg='/',
            max=100,
            bg_colour=0xffffff,
            bg_alpha=0.2,
            fg_colour=0xffffff,
            fg_alpha=0.3,
            x=165, y=170,
            radius=135,
            thickness=50,
            start_angle=-120,
            end_angle=120
      },
}

require 'cairo'

function rgb_to_r_g_b(colour,alpha)
      return ((colour / 0x10000) % 0x100) / 255., ((colour / 0x100) % 0x100) / 255.,
(colour % 0x100) / 255., alpha
end
```

```
function draw_ring(cr,t,pt)
	local w,h=conky_window.width,conky_window.height

	local
xc,yc,ring_r,ring_w,sa,ea=pt['x'],pt['y'],pt['radius'],pt['thickness'],pt['start_angl
e'],pt['end_angle']
	local bgc, bga, fgc, fga=pt['bg_colour'], pt['bg_alpha'], pt['fg_colour'],
pt['fg_alpha']

	local angle_0=sa*(2*math.pi/360)-math.pi/2
	local angle_f=ea*(2*math.pi/360)-math.pi/2
	local t_arc=t*(angle_f-angle_0)

	-- Dessine le fond du cercle

	cairo_arc(cr,xc,yc,ring_r,angle_0,angle_f)
	cairo_set_source_rgba(cr,rgb_to_r_g_b(bgc,bga))
	cairo_set_line_width(cr,ring_w)
	cairo_stroke(cr)

	-- Dessine le premier plan du cercle

	cairo_arc(cr,xc,yc,ring_r,angle_0,angle_0+t_arc)
	cairo_set_source_rgba(cr,rgb_to_r_g_b(fgc,fga))
	cairo_stroke(cr)
end

function conky_ring_stats()
	local function setup_rings(cr,pt)
		local str=''
		local value=0

		str=string.format('${%s %s}',pt['name'],pt['arg'])
		str=conky_parse(str)

		value=tonumber(str)
		pct=value/pt['max']

		draw_ring(cr,pct,pt)
	end

	if conky_window==nil then return end
	local
cs=cairo_xlib_surface_create(conky_window.display,conky_window.drawable,conky_window.
visual, conky_window.width,conky_window.height)

	local cr=cairo_create(cs)

	local updates=conky_parse('${updates}')
	update_num=tonumber(updates)

	if update_num>5 then
		for i in pairs(settings_table) do
			setup_rings(cr,settings_table[i])
		end
	end
end
```

Pour tout vous avouer , je ne comprends absolument rien au lua :D et ce n'est pas le sujet de ce wiki, mais ce sript est si bien fait qu'il vous suffit de remplir les champs **correctement** et de suivre

les instructions des archives téléchargées, pour obtenir toutes sortes de résultats:


saturday 17
july 2010
07:22
6%
28%
97%
24%
0%
0%
Sat Jul 17, 7:22 PM

1: http://londonali1010.deviantart.com/art/My-Conky-Config-311209-148712243 par londonali1010.

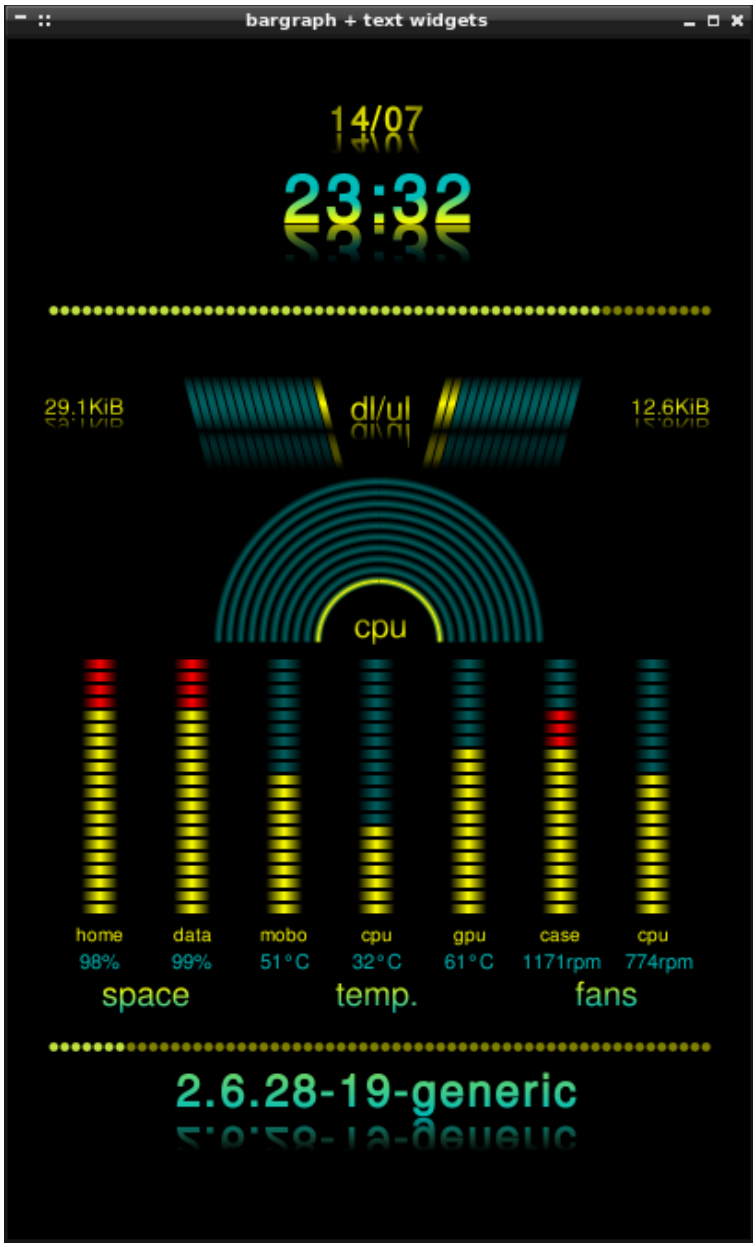
2: http://jpope777.deviantart.com/art/Conky-Rings-3-139828675 par jpope777.

3: http://g0rg0d.deviantart.com/art/Cornet-Conky-171920478 par g0rg0d.

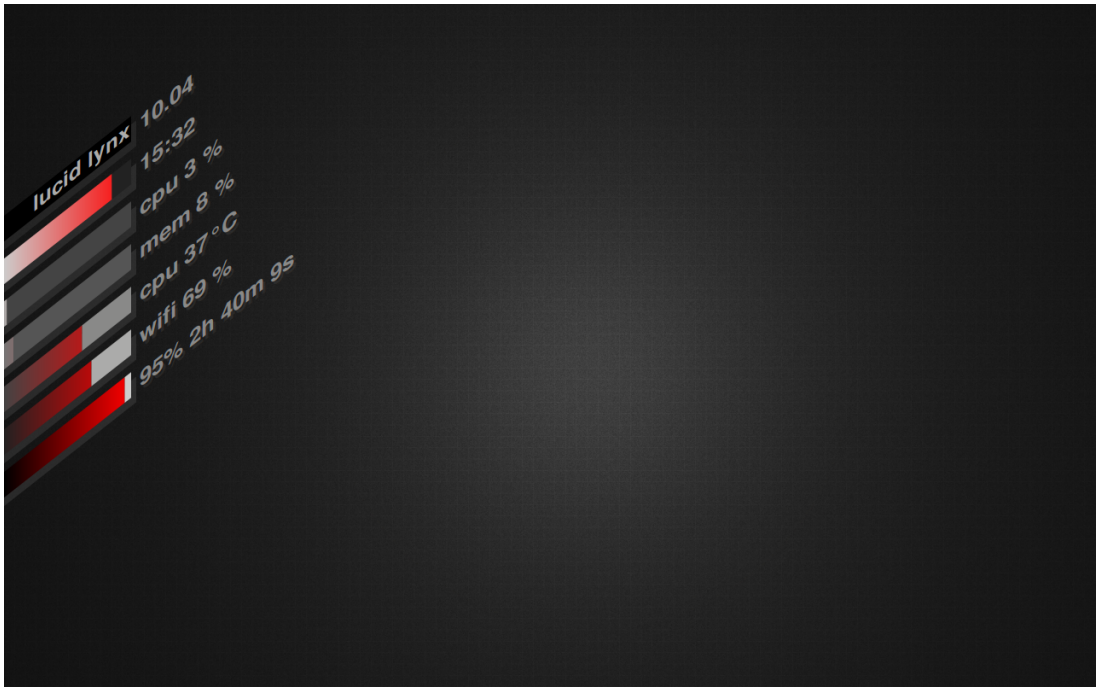
4: http://sunjack94.deviantart.com/art/Electric-Conky-204993834 par sunjack94.

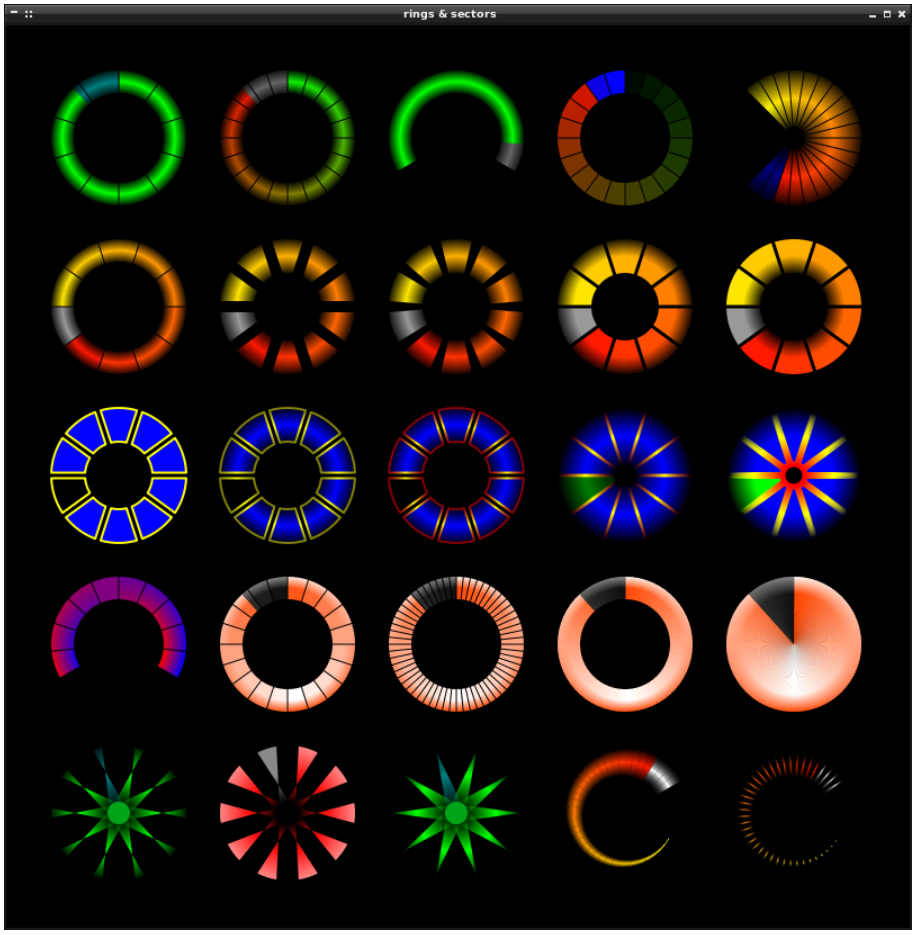
5: http://arpinux.deviantart.com/art/more-conky-rings-139488211 par arpinux.

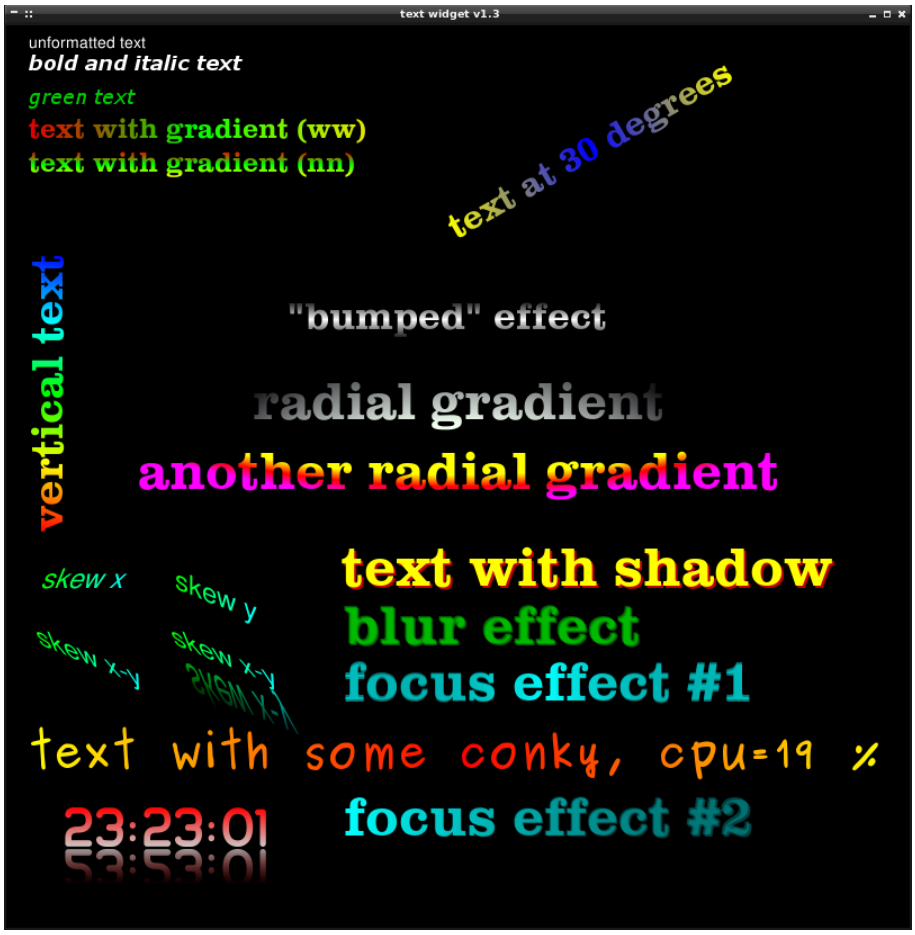
Londonali1010 a également publié un script “widgets.lua” qui permet d'inclure des “morceaux” de code, les “widgets”. L'utilisation de ce genre de widget permet des conkys de ce type:


bargraph + text widgets
14/07
23:32
29.1KiB
dl/ul
12.6KiB
cpu
home
98%
data
99%
mobo
51°C
cpu
32°C
gpu
61°C
case
1171rpm
cpu
774rpm
space
temp.
fans
2.6.28-19-generic


Nirvana
Nervermind
Something In The Way


lucid lynx 10.04
15:32
cpu 3 %
mem 8 %
cpu 37°C
wifi 69 %
95% 2h 40m 9s





tous ces widgets ont été écrit par wlourf :)

Conky Weather

Alors là c'est super facile:

- ouvrez la fenêtre... non, pas sur le bureau, la **Vraie**
- ouvrez les yeux et regardez dehors. :)

Pour les prévisions, on a les proverbes populaires ou les fabuleux scripts de kaivalagi ;).

Conclusion et Recommandations

Vous pouvez désormais personnaliser votre bureau à votre guise grâce à **conky**. Il existe quelques règles élémentaires lorsque vous récupérez un conkyrc/script.lua sur le net:

- toujours éditer les fichiers concernés et vérifier **chaque** variable. une erreur de variable ou de fonction peut conduire à une clôture prématurée du programme.
- afin de tester votre configuration, je vous conseille de lancer votre conky depuis un terminal avec une commande du genre:

```
conky -c ~/chemin_de_votre_conky
```

Ce qui vous permettra de visualiser les erreurs possibles et d'y remedier avant d'inclure votre conky dans votre ~/.config.openbox/autostart.sh.

- les variables “$exec” “$execp” sont gourmandes en ressources, préférez les appels aux scripts externes avec une commande du genre:

```
..${texeci 120 ~/bin/mon_script_bash}...
..${texeci 120 phython ~/bin/mon_script_python}...
```

Ce qui permettra à conky de s'afficher sans attendre le résultat du script. L'affichage se mettra à jour dès l'arrivée des informations.

ce wiki ainsi que les fonctions et variables décrites concernent conky_1.7.2 et ultérieur.

Liens

la page du projet sur sourceforge : http://conky.sourceforge.net/

Conky Pitstop (maintenu en français par wlourf) : http://conky.pitstop.free.fr/wiki/index.php5?title=Main_page_(fr)

Le conky-artists-group sur deviantart: http://conky-artists-group.deviantart.com/

Conky : les fonctions

Les fonctions passées à conky déterminent sa position, ses dimensions, la police de caractères utilisée... bref, comment s'affiche conky. Cette page est une traduction (à compléter) de celle du manpage de Conky_1.8.0.

Fonctions	**Arguments**	**Définitions**
alignment	top_left,top_right,top_middle, bottom_left,bottom_right,bottom_middle, middle_left,middle_middle,middle_right.	position de conky sur l'écran.
apend_file	adresse_du_fichier	prend en compte un fichier: les fonctions seront rejoutées à celles du conky en cours.
background	yes ou no	si yes, conky sera lancé en “background”: il vous rendra la main.
border_inner_margin	nombre (pixels)	espace en pixels entre le texte et la bordure.
border_outer_margin	nombre (pixels)	espace en pixels entre la bordure et le bord de la fenêtre.
border_width	nombre (pixels)	épaisseur de la bordure en pixels.
colorN(0→9)	hexadecimal “XXXXXX” sans “#“	couleur prédéfinie (de 0 à 9) à utiliser dans la zone **TEXT**.
cpu_avg_samples	nombre	échantillon à prendre en compte pour le monitoring du cpu.
default_bar_size	nombre(pixels) hauteur longueur	dimensions par défaut des barres, ex: 'default_bar_size 4 10' très utile pour les fonctions execbar et execibar.
default_color	hexadécimal/couleur	couleur par défaut du texte et des bordures, ex: 'default_color ffffff' ou 'default_color white'.
default_gauge_size	(pixels) hauteur longueur	dimensions par défaut des compteurs, ex 'default_gauge_size 15 15' très utile pour les fonctions execgauge et execigauge.
default_graph-size	(pixels) hauteur longueur	dimensions par défaut des graphiques, ex 'default_graph-size 4 10' très utile pour les fonctions execgraph et execigraph.
default_outline_color	hexadécimal/couleur	couleur par défaut du contour de la police.
default_shade_color	hexadécimal/couleur	couleur par défaut de l'ombre de la police et des graphiques.
disable_auto_reload	yes ou no	si yes, conky ne recharge pas son fichier de configuration après une modification.
diskio_avg_samples	nombre	échantillon à prendre en compte pour le monitoring du la lecture/écriture.

display	–	choix de l'écran pour l'affichage.
double_buffer	yes ou no	si yes, utilise l'extension Xdbe: permet d'éliminer le clignotement. Il est recommandé d'utiliser une fenêtre propre pour Conky afin de limiter la taille du double tampon.
draw_borders	yes ou no	si yes, conky affiche une bordure autour du texte.
draw_graph_borders	yes ou no	si yes, conky dessine une bordure autour des graphiques.
draw_outline	yes ou no	si yes, conky utilise la police avec un contour.
draw_shades	yes ou no	si yes, conky affiche les ombres sous le texte et les graphiques.
extra_newline	yes ou no	si yes et si 'out_to_console yes', conky rajoute une ligne en fin de fichier. Utile pour “piper” conky dans une wibox d'awesome.
font	nom_de_la_police	détermine la police utilisée par conky. *obsolète»voir xftfont*.
format_human_readable	yes ou no	si yes, conky tente d'afficher les valeurs dans un format lisible (Kib,Mib,Gib). si no, conky affiche toutes les valeurs en bytes.
gap_x	nombre (pixels)	marge en pixels entre le bord de l'écran et le coté droit/gauche de la fenêtre conky.
gap_y	nombre (pixels)	marge en pixels entre le bord de l'écran et le haut/bas de la fenêtre conky.
hddtemp_host	XXX.X.X.X	hôte sur lequel se connecte hddtemp, défaut: 127.0.0.1
hddtemp_port	–	port utilisé par hddtemp, défaut: 7634
if_up_sctrictness	up, link, address	détermine comment conky évalue la connexion réseau.
imap	serveur gloal IMAP; arguments : “host user pass [-i interval] [-f folder] [-p port] [-e command]”. Port par défaut: 143, dossier par défaut: 'INBOX', intervalle par défaut: 5 minutes. Si le mot de passe est '*', il vous sera demandé au démarrage de Conky.	
imlib_cache_flush_interval	nombre(secondes)	intervalle de rafraichissement du cache imlib en secondes.
imlib_cache_size	nombre (bytes)	taille du cache imlib, par défaut: 4MiB. Si conky affiche de nombreuses images, il est conseillé de le désactiver avec 'imlib_cache_size 0'.
lua_draw_hook_post	nom_de_la_fonction	Fix Me!
lua_draw_hook_pre	nom_de_la_fonction	Fix Me!
lua_load	adresse du/des script(s)	conky charge le(s) script(s) lua, ex: 'lua_load ~/.conky/clock.lua'.
lua_shutdown_hook	nom_de_la_fonction	vide le cache de la 'fonction' afin préserver le système à l'extinction.
lua_startup_hook	nom_de_la_fonction	Fix Me!

mail_spool	–	mail spool pour la vérification des mails.
max_port_monitor_connections	nombre	nombre maximum de scan par port, par défaut 256.
max_specials	nombre	nombre maximal de variables spéciales; fonts, offsets, aligns... par défaut 512.
max_user_text	nombre (bytes)	taille maximale du buffer utilisateur, par défaut 16384 bytes.
maximum_width	nombre (pixels)	largeur maximale de la fenêtre conky en pixels.
minimum_size	(pixels) largeur hauteur	dimension minimale de la fenêtre conky, ex: 'minimum_size 200 500'.
mpd_host	–	hôte pour mpd *music player daemon*.
mpd_password	–	mot de passe du serveur mpd.
mpd_port	–	port pour mpd.
music_player_interval	nombre (seconds)	intervalle de vérification des données des lecteurs audio, par défaut: celui de conky.
net_avg_samples	nombre	échantillon à prendre en compte pour le monitoring du net.
no_buffers	yes ou no	si oui, conky ne prend pas en compte le buffer dans le calcul de l'utilisation mémoire.
out_to_console	yes ou no	si yes, envoi des informations de conky au STDOUT. Utile pour “piper” conky dans une statusbar.
out_to_ncurses	yes ou no	si yes, comme 'out_to_console' mais avec un rafraichissement de l'affichage au lieu d'une succession d'informations.
out_to_stderr	yes ou no	si yes, envoi des informations de conky au STDERR.
out_to_x	yes ou no	si no, pas d'affichage dans X: vérifier qu'aucune fonction graphique ne soit enclenchée, et vérifier l'envoi des informations vers STDOUT ou STDERR.
override_utf8_locale	yes ou no	si yes, conky force l'encodage local utf-8. requiert XFT.
overwrite_file	adresse_du_fichier	écrase le fichier donné en argument.
own_window	yes ou no	si yes, conky dessine sa propre fenêtre.
own_window_class	nom 'WM_CLASS'	determine manuellement le 'WM-CLASS' de conky, par défaut: Conky.
own_window_color	hexadécimal/couleur	couleur de la fenêtre conky, ex: 'own_window_color 000000' ou 'own_window_color black'.
own_window_hints	undecorated,below,above,sticky, skip_taskbar,skip_pager.	si 'own_window yes', configure les propriétés de conky dans le WM. si 'own_window_type desktop', certaines valeurs sont définies par défaut. si 'own_window_type override', ces valeurs sont ignorées.
own_window_title	titre	determine manuellement le titre de la

		fenêtre conky, par défaut: '<hôte> - conky'.
own_window_argb_visual	yes ou no	ARGB est utilisé pour les effets de transparence. Notez que vous aurez besoin d'un gestionnaire composite pour cela. Cette option fonctionne avec la fonction 'own_window_type override'.
own_window_argb_value	nombre (0→255)	niveau de transparence pour conky, où 0=transparent et 255=opaque. Notez que si 'own_window_transparent yes', cette fonction n'a pas d'effet.
own_window_transparent	yes ou no	si yes, conky est transparent: comme 'own_window_argb_value 0'.
own_window_type	normal,desktop,dock,panel,override	si '**own_window yes**', determine le type de fenêtre utilisé par conky, par défaut: '**normal**': conky est traité comme les autre fenêtres et est controlé par le WM. '**desktop**': aucune décoration, toujours visible, pas dans la barre des taches, visible sur tous les bureaux virtuels. '**dock**': conky est placé dans le dock et en prend les propriétés, utile pour les wm possédant cette option (*box, pekwm..). '**panel**': conky adopte les propriétés d'une barre de tache: pas de décoration, toujours visible, non-recouvrable par les fenêtres maximisées, visible sur tous les bureaux. '**override**': conky n'est pas contrôlé par le WM, les 'hints'(conseils) sont ignorés. Cette option peut être utile pour les tilingWM (dwm,awesome,xmonad...).
pad_percents	Fix Me!	
pop3	serveur global POP3; arguments : “host user pass [-i interval] [-p port] [-e command]”. Port par défaut : 110, intevalle par défaut : 5 minutes. Si le mot de passe est '*', il vous sera demandé au démarrage de Conky.	
short_units	yes ou no	réduit l'affichage des unités: kiB=k, GiB=G, ... 'no' par défaut.
show_graph_range	yes ou no	si yes, conky affiche l'amplitude des graphiques. 'no'par défaut.
show_graph_scale	yes ou no	si yes, affiche la valeur maximale dans les graphiques.
stippled_borders	nombre (pixels)	si 'draw_borders yes', taille en pixels des pointillés de la bordure de conky.
temperature_unit	farenheit ou celsius	détermine les unités de température, par défaut: celsius.
templateN	nombre (0→9)	determine un modèle à utiliser après TEXT. Fix Me! expliquer la syntaxe des modèles.
text_buffer_size	nombre (bytes)	taille du buffer utilisé par conky, par défaut/minimum: 256. Ce tampon est utilisé pour stocker les informations renvoyées par les variables. tester plusieurs valeurs pour

		choisir entre la quantité de variables affichées et les performances de conky.
times_in_seconds	yes ou no	si yes, les variables de temps sont affichées en secondes. cette fonctions n'affecte pas $time, $tztime, ou $utime.
top_cpu_separate	yes ou no	si yes, la variable 'top' renvoie l'usage d'un processeur. si no, 'top' renvoi l'usage des processeurs combinés.
top_name_width	nombre	longueur des noms des processus renvoyés par 'top', par défaut: 15.
total_run_times	nombre	nombre de rechargements de conky avant de quitter. Zéro fait tourner conky indéfiniment.
update_interval	nombre (secondes)	fréquence de rafraichissement de conky.
update_interval_on_battery	nombre (secondes)	fréquence de rafraichissement de conky sur battery.
uppercase	yes ou no	si yes, affiche les caractères en majuscules.
use_spacer	left,right,none	ajoute un espace avant ou après certaines variables pour les empêcher de bouger. **ne fonctionne qu'avec une police à chasse fixe**.
use_xft	yes ou no	si yes, conky utilise les polices 'xft', active l'anti-crenelage...
xftalpha	nombre (0→1)	opacité de la police définie par 'xftfont', 0=transparent, 1=opaque.
xftfont	nom_de_la_police:size=nombre	si '**use_xft yes**', détermine la police par défaut utilisée par conky. ex: 'xftfont Terminus:pixelsize=10' - 'xftfont Aller:size=8:bold'.
TEXT	ici commence l'affichage des variables.	

Conky : les variables

Les variables passées à conky déterminent les informations affichées. Cette page est une traduction de celle du manpage de conky.

Certains passages sont empruntés à l'excellent post de nonas sur pcimpact.

Variables	Arguments	Définitions
acpiacadapter	-	indique l'état du chargeur acpi: branché/débranché.
acpifan	-	indique les infos du capteur acpi pour le ventilateur.
acpitemp	-	indique la temperature captée par acpi.
addr	interface	adresse IP pour l'interface désignée, ex: '${addr eth0}'.
addrs	interface	comme 'addr' pour linux seulement.
adt746xcpu	-	température du capteur therm_adt746x.
adt746xfan	-	vitesse du ventilateur captée par therm_adt746x.
alignc	nombre ('n'pixels)	alignement du texte au centre de la ligne, avec un décallage de 'n', ex: '${alignc 10}'.
alignr	nombre ('n'pixels)	alignement du texte à droite de la ligne, avec un décallage de 'n', ex: '${alignr 5}'.
section apcups* - ces variables affiches les informations délivrées par votre matériel APC™ .		
apcupsd	hôte:port	lance le démon de controle des fonctions APC, par défaut 'localhost:3551'. équivalent libre à powerchute.
apcupsd_cable	-	affiche le type de connexion UPS.
apcupsd_charge	-	affiche le poucentage de charge de la batterie courante.
apcupsd_lastxfer	-	affiche la raison de la dernière bascule de 'ligne' à 'batterie'.
apcupsd_linev	-	affiche l'entrée nominale en volt.
apcupsd_load	-	affiche le taux de charge courant.
apcupsd_loadbar	-	affiche une barre de progression de la charge courante.
apcupsd_loadgauge	hauteur,largeur	affiche un compteur de progression de la charge courante.
apcupsd_loadgraph	hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression de la charge courante. ex: '${apcupsd_loadgraph 20,120 FFFFFF 000000}'.
apcupsd_model	-	affiche le modèle UPS.
apcupsd_name	-	affiche le nom défini par utilisateur.
apcupsd_status	-	affiche le statut courant : 'en ligne' ou 'sur batterie'.
apcupsd_temp	-	affiche la température de l'UPS.
apcupsd_timeleft	-	affiche le temps restant sur la batterie.
apcupsd_upsmode	-	affiche le 'mode' UPS, ex: 'indépendant'.
fin de section apcups* .		
apm*	*FreeBSD only*	
section audacious* - ces variables renvoient les informations délivrées par le lecteur audio		

***audacious*.**		
audacious_bar	hauteur,longueur	affiche une barre de progression du titre en cours dans audacious.
audacious_bitrate	-	affiche le debit du morceau en cours.
audacious_channels	-	affiche le numéro du canal audio utilisé par audacious.
audacious_filename	-	affiche l'adresse complète du morceau en cours.
audacious_frequency	-	affiche la fréquence du morceau en cours.
audacious_length	-	affiche la durée du morceau en cours sous la forme MM:SS .
audacious_length_seconds	-	affiche la durée du morceau en cours en secondes.
audacious_main_volume	-	affiche le volume individuel d'audacious.
audacious_playlist_length	-	affiche le nombre de morceaux dans la liste de lecture en cours.
audacious_playlist_position	-	affiche la position du morceaux dans la liste de lecture en cours.
audacious_position	-	affiche la progression du morceau en cours sous la forme MM:SS .
audacious_position_seconds	-	affiche la progression du morceau en cours en secondes.
audacious_status	-	affiche le statut d'audacious, 'en lecture','en pause',stoppé','aucune lecture'.
audacious_title	nombre	affiche le titre du morceau en cours. une longueur maximale peut être passé en argument.
***fin de section audacious* .**		
battery	-	affiche le statut de la batterie et la charge en pourcentage, ex: '${battery BAT0}'.
battery_bar	hauteur,longueur nombre	affiche une barre de progression de la charge de la batterie, ex: ${battery_bar 4,40 BAT0}'.
battery_percent	nombre	affiche la charge de la batterie en pourcentage, ex: '${battery BAT0}%'.
battery_short	nombre	affiche le statut de la batterie en format court et la charge en pourcentage.
battery_time	nombre	temps restant avant la (de)charge complète de la batterie.
blink	'text' ou 'variables'	fais clignoter 'text' et/ou 'conky_variables' au rhytme de 'update_interval'.
***section bpmx* - ces variables renvoient les informations délivrés par la lecteur *bmpx*.**		
bmpx_album	-	affiche l'album du morceau en cours.
bmpx_artist	-	affiche l'artiste du morceau en cours.
bmpx_bitrate	-	affiche le debit du morceau en cours.
bmpx_title	-	affiche le tritre du morceau en cours.
bmpx_track	-	affiche le numéro du morceau en cours.
bmpx_uri	-	affiche l'adresse internet du morceau en cours.
***fin de section bmpx* .**		
buffers	nombre (bytes)	taille de la mémoire tampon.

cached	nombre (bytes)	taille de la mémoire en cache.
cmdline_to_pid	expression	pid du premier processus ayant 'expression' dans sa ligne de commande.
color	#hexadécimal/couleur	change la couleurdu texte qui suit, ex: '${color #01AB2C}${cpu cpu0}%${color}'. notez la variable '${color}' en fin de ligne pour rétablir la couleur par défaut.
colorN	-	change la couleur du texte qui suit en utilisant un couleur prédéfinie par la fonction 'colorN'. ex: '${color1}${cpu cpu0}%${color2}${cpugraph cpu0 6,40}${color}'.
combine	variable1 variable2	place la variable2 à droite de la variable1, séparée par le caractère inscrit. ex: '${combine ${head /proc/cpuinfo 2} :: ${head /proc/meminfo 1}}'. affiche : en ligne 1; “cpuinfo_ligne1 :: meminfo_ligne1” et en ligne 2; “cpuinfo_ligne2”.
conky_build_arch	-	architecture du processeur sur lequel conky a été construit.
conky_build_date	-	date à laquelle conky a été construit.
conky_version	-	version du conky utilisé.
cpu	cpux	utilisation du/des processeur(s) 'x' en pourcentage, '${cpu}' ou '${cpu cpu0}', affiche l'utilisation totale des processeurs, '${cpu cpux}' ou x>=1, affiche l'utilisation individuelle des processeurs.
cpubar	cpux hauteur,longueur	affiche une barre d'état pour l'utilisation du/des processeur(s), ex: '${cpubar cpu0 10,60}'.
cpugauge	cpux hauteur,largeur	affiche un compteur d'utilisation du/des processeurs 'x', ex: '${cpugauge cpu2 25,25}'.
cpugraph	hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression de l'usage du/des processeurs.
curl	url temps (minutes)	récupère des données d'une adresse 'url' à un intervalle 'temps', par défaut: 15 minutes.
desktop	-	numéro du bureau sur lequel conky est affiché.
desktop_name	-	nom du bureau sur lequel conky est affiché.
desktop_number	-	nombre total de bureaux sur lequel conky peut s'afficher.
disk_protect	disque	état de protection du disque 'disque' si supporté par le noyau.
diskio	disque	affiche la quantité de données en lecture/écriture sur le disque 'disque', ex: '${diskio sda3}/s'.
diskio_read	disque	affiche la quantité de donnée en lecture sur le disque 'disque', ex: '${diskio_read sda}/s'.
diskio_write	disque	affiche la quantité de donnée en écriture sur le disque 'disque', ex: '${diskio_write sda4}/s'.

diskiograph	disque hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression des données en lecture/écriture sur le disque 'disque'. ex: '${diskiograph /dev/sda3 15,70 ffffff 000000 -t}'.
diskiograph_read	disque hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression des données en lecture sur le disque 'disque'. ex: '${diskiograph_read /dev/sda1 15,70 ffffff 000000 -t}'.
diskiograph_write	disque hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression des données en écriture sur le disque 'disque'. ex: '${diskiograph_write /dev/sda 15,70 ffffff 000000 -t}'.
downspeed	interface	affiche la quantité de données téléchargées pour 'interface', ex: '${downspeed eth0}/s'.
downspeedf	interface	affiche la quantitée de données téléchargées en KiB pour 'interface'.
downspeedgraph	interface hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique des données téléchargées pour 'interface', ex: '${downspeedgraph eth0 10,60 FFFFFF 000000}'.
draft_mails	maildir intervalle	affiche le nombre de mails marqué comme brouillons dans 'maildir'.
else	-	texte à afficher si la condition “if” n'est pas valide, ex: '${if_up eth0}réseau ethernet disponible${else}pas de réseau ethernet${endif}'.
endif	-	fini un bloc 'if'/'else'.
entropy_avail	-	Fix Me!
entropy_bar	hauteur,longueur	Fix Me!
entropy_perc	-	Fix Me!
entropy_poolsize	-	Fix Me!
eval	expression	Fix Me!
eve	api_userid api_key character_id	affiche des informations relative au profil du 'userid' venant du jeu Eve Online.
section “exec” - préférez la commande '$execi' pour préserver vos ressources. voir aussi “texec*”		
exec	commande	execute une commande au rhytme de 'update_interval',et affiche son résultat dans conky, ex: '${exec cat ~/todo | grep important}'.
execbar	commande	affiche une barre d'état de la 'commande' si celle-ci renvoie un résultat entre 0et100. les dimensions de la barre peuvent être réglées grâce à 'default_bar_size'.
execgauge	commande	affiche un compteur d'état de la 'commande' si celle-ci renvoie un résultat entre 0et100. les dimensions du compteur peuvent être géglées grâce à 'default_gauge_size'.
execgraph	-t -l commande	affiche un graphique d'état de la 'commande' si celle-ci renvoie un résultat entre 0et100. les dimensions du graphique peuvent être réglées grâce à

		'default_graph_size'. si le premier argument est '-t' ou '-l', pensez à le faire précéder d'un espace ' '.
execi	intervalle commande	idem que 'exec' mais la 'commande' se lance à un 'intervalle' spécifique. l''intervalle' doit être >= 'update_interval'. voir aussi '$texeci'.
execibar	intervalle commande	idem que 'execbar' mais la 'commande' se lance à un 'intervalle' spécifique.
execigauge	intervalle commande	idem que 'execgauge' mais la 'commande' se lance à un 'intervalle' spécifique.
execigraph	-t -l commande	idem que 'execgraph' mais la 'commande' se lance à un 'intervalle' spécifique.
execp	commande	éxécute une commande et affiche son résultat dans conky. accepte la syntaxe de conky. vous pouvez inclure des expressions telles que '${color1}text${color}' dans votre 'commande'.
execpi	intervalle commande	idem que 'execp' mais la 'commande' se lance à un 'intervalle' spécifique.
fin de la section exec*.		
flagged_mails	maildir intervalle	affiche le nombre de mails 'marqués' dans 'maildir'.
font	nom_de_la_police:size=nombre	spécifie une police particulière pour le texte qui suit, ex: '${font Terminus:pixelsize=10}${color1}heure:${time %I:%M}${font}${color}'. notez que '${font}' restaure la police par défaut.
format_time	secondes format	si 'times_in_seconds yes', affiche le temps donné en secondes selon un 'format'. Fix Me!
forwarded_mails	maildir intervalle	affiche le nombre de mails marqués comme 'transférés' dans 'maildir'.
freq	n	affiche la fréquence du processeur 'n'>=1 en MHz.
freq_g	n	affiche la fréquence du processeur 'n'>=1 en GHz.
fs_bar	hauteur,longeur fs	affiche une barre d'état pour l'occupation de 'fs', ex: '${fs_bar /home/arp 6,20}'.
fs_bar_free	hauteur,longeur fs	affiche une barre d'état pour l'espace libre sur 'fs'.
fs_free	fs	affiche l'espace disponible sur 'fs', ex: ${fs_free /media/backup/}'.
fs_free_perc	fs	affiche le pourcentage d'espace disponible sur 'fs'.
fs_size	fs	affiche la taille de 'fs'.
fs_type	fs	affiche le type de système de fichiers de 'fs' (ext2,ext3,ext4,vfat,ntfs..).
fs_used	fs	affiche l'espace utilisé sur 'fs'.
fs_used_perc	fs	affiche le pourcentage d'espace utilisé sur 'fs'.
goto	x	déplace le prochain élément au pixel 'x'.
gw_iface	-	affiche l'interface du router ou 'multiple' ou 'aucun' selon l'état.
gw_ip	-	affiche l'IP de la passerelle par défaut ou 'multiple' ou 'aucune' selon l'état.

hddtemp	disque	affiche la température du 'disque'. nécessite que le démon hddtemp soit lancé et accessible par l'utilisateur.
head	fichier nombre check	affiche les 'n'(nombre) premières lignes du 'fichier' spécifié. l'intervalle de vérification est défini par 'check' en secondes, par defaut , check=2sec.
hr	nombre ('n'pixels)	trace une ligne horizontale d'épaisseur 'n'
hwmon	dev type n factor offset	affiche les informations transmises par la sonde hwmon de sysfs (Linux2.6). 'dev' peut être ignoré si vous n'avez qu'un seul capteur. le type peut être 'in'/'vol' pour voltage, 'fan' ou 'temp'. 'n' est le numéro de la sonde (voir /sys/class/hwmon).'factor' et 'offset' sont optionnels et autorisent le recalcul des données entrantes.
i2c	dev type n factor offset	affiche les informations transmises par la sonde i2s de sysfs (Linux2.6). 'dev' peut être ignoré si vous n'avez qu'un seul capteur. le type peut être 'in'/'vol' pour voltage, 'fan' ou 'temp'. 'n' est le numéro de la sonde (voir /sys/bus/i2c/devices/). 'factor' et 'offset' sont optionnels et autorisent le recalcul des données entrantes.
section ik8* - ces variables sont spécifiques aux ordinateurs portables Dell™ Inspiron. necessite le driver iK8.		
i8k_ac_status	-	affiche le statut de la ligne (branché ou non) transmit par /proc/ik8.
i8k_bios	-	affiche la version du bios listée dans /proc/ik8.
i8k_buttons_status	-	affiche le statut de touches listeés dans /proc/ik8.
i8k_cpu_temp	-	affiche la température du processer en celsius trouvée dans /proc/ik8.
i8k_left_fan_rpm	-	affiche la vitesse de rotation du ventilateur de gauche trouvée dans /proc/ik8.
i8k_left_fan_status	-	affiche le statut du ventilateur de gauche trouvé dans /proc/ik8.
i8k_right_fan_rpm	-	affiche la vitesse de rotation du ventilateur de droite trouvée dans /proc/ik8.
i8k_right_fan_status	-	affiche le statut du ventilateur de droite trouvé dans /proc/ik8.
i8k_serial	-	affiche le numéro de série de votre ordinateur trouvé dans /proc/ik8.
i8k_version	-	affiche la version de votre sonde ik8.
fin de la section ik8*.		
ibm_brightness	-	affiche la luminosité de l'écran des ordinateurs portables Ibm™.
ibm_fan	-	affiche la vitesse de rotation du ventilateur des ordinateurs portables Ibm™.
ibm_temps	nombre (0→7)	affiche les températures 'n' des ordinateurs portables Ibm™. ex: 'T°cpu = ${ibm_temps 0}°C'. cpu:O, hdd:2, gpu:3 .
ibm_volume	-	affiche le volume interne des ordinateurs portables Ibm™ (de 0→14).
iconv_start	encodage entrée/sortie	convertiseur d'encodage, utilise iconv, doit être stoppé avec 'iconv_stop'.

iconv_stop	-	arrêt de la conversion de l'encodage enclenchée par 'iconv-start'.
section "if" - fonctionne sous la forme "$if*var0 text1/var1 $else text2/var2 $endif" .		
if_empty	variable	si la 'variable' est vide, affiche tout entre '$if_empty' et son '$endif'.
if_existing	fichier expression	si le 'fichier' existe, affiche tout entre '$if_exist' et son '$endif'. le second paramêtre est optionnel et rajoute une précision sur le fichier.
if_gw	-	si il existe au moins une passerelle, affihe tout entre '$if_gw' et son '$endif'.
if_match	expression	si 'expression' remplit les conditions, affiche tout entre '$if_match' et son '$endif'. ex: '${if_match "${ibm_temps 0}" >= "80" }${color red}${ibm_temps 0}°C${else}${color green}${ibm_temps 0}°C${endif}'.
if_mixer_mute	mixer	si le 'mixer' est en sourdine, affiche tout entre '$if_mixer_mute' et son '$endif', par défaut mixer=Master.
if_mounted	point_de_montage	si le 'point_de_montage' est monté, affiche tout entre '$if_mounted' et son '$endif'.
if_mpd_playing	-	si le serveur mpd est en lecture ou en pause, affiche tout entre '$if_mpd_playing' et son '$endif'.
if_running	processus	si le 'processus' est en cours, affiche tout entre '$if_running' et son '$endif'.
if_smapi_bat_installed	index	si 'smapi' est installé sur 'index', affiche tout entre '$if_smapi_bat_installed' et son '$endif'. voir 'smapi_*'.
if_up	interface	si 'interface' est active, affiche tout entre '$if_up' et son '$endif'. voir aussi 'if_up_strictness'.
if_updatenr	nombre d'updates 'n'	si conky s'est relancé 'n' fois, affiche tout entre '$if_updatenr' et son '$endif'.
if_xmms2_connected	-	si xmms2 tourne, affiche tout entre '$if_xmms2_connected' et son '$endif'.
fin de la section "if".		
image	adresse -p x,y -s LxH -n -f intervalle	affiche l'image selon son 'adresse' à la position -p ou 'x' est le décallage horizontal en pixels depuis le coin supérieur gauche de conky, et 'y' le décallage vertical. possibilité de '-s' redimensionner l'image en indiquant sa 'L'argeur et sa 'H'auteur, de ne pas mettre l'image en cache '-n', ou de vider le cache '-f' à un 'intervalle' déterminé.
imap_messages	affiche le nombre de message dans votre boîte mail IMAP (voir imap). vous pouvez définir des boîtes IMAP individuelles avec les arguments suivants: "host user pass [-i interval] [-p port] [-e command]". Port par défaut : 110, intervalle par défaut : 5 minutes. Si le mot de passe est '*', il vous sera demander au démarrage de Conky.	
imap_unseen	-	idem que 'imap_messages' pour les nauveaux messages.
include	adresse	prend en compte le ficher indiqué par 'adresse', les fonctions se rajoutent aux fonctions du conky en cours, les avariables se placent à l'endroit où 'include' est lancé.
ioscheduler	disque	affiche l'ordonnancement des entées/sorties pour le

		'disque'.voirici.
kernel	-	affiche la version du kernel utilisé.
laptop_mode	-	retourne les valeurs de /proc/sys/vm/laptop_mode
lines	fichier	affiche le nombre de lignes contenues dans le 'fichier'.
loadavg	1,2,3	affiche la charge moyenne processeur; 1=dernière minute, 2=dernières 5 minutes, 3=dernières 15 minutes.
loadgraph	hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de progression de la charge moyenne système.
lua	fonction paramêtres	éxécute une 'fonction' lua selon ses 'paramêtres' et affiche le résultat.
lua_bar	hauteur,longueur fonction paramêtres	éxécute une 'fonction' lua selon ses 'paramêtres' et affiche le résultat (compris entre 1→100) dans une barre.
lua_gauge	hauteur,largeur fonction paramêtres	éxécute une 'fonction' lua selon ses 'paramêtres' et affiche le résultat (compris entre 1→100) dans un compteur.
lua_graph	fonction paramêtres couleur1 couleur2 scale -t -l	éxécute une 'fonction' lua selon ses 'paramêtres' et affiche le résultat (compris entre 1→100) dans un graphique.
lua_parse	fonction paramêtres	Fix Me!
machine	-	affiche l'architecture du noyau, ex: i686.
mails	boîte_mail intervalle	affiche le nombre de messages de votre 'boîte_mail' mbox et maildir supportés.
mboxscan	-n 'nombre_de_messages' -fw 'longueur du mail' -sm 'longueur_du_sujet' mbox	affiche un résumé des 'n' derniers messages de 'mbox', ex: '${mboxscan -n 10 “/home/arp/mail box”}'.
mem	-	affiche la quantité de mémoire vive utilisée.
membar	hauteur,longueur	affiche une barre d'état de l'utilisation de la mémoire vive.
memeasyfree	-	affiche la quantité de mémoire vive disponible, en incluant buffer/cache.
memfree	-	affiche la quantité de mémoire vive disponible.
memgauge	hauteur,largeur	affiche un compteur d'état de la quantité de mémoire vive utilisée.
memgraph	hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique de la progression d'utilisation de la mémoire vive.
memmax	-	affiche la quantité maximale de mémoire vive.
memperc	-	affiche le pourcentage d'utilisation de la mémoire vive.
mixer	sortie	affiche le volume de 'sortie' du mixer, par défaut sortie=Master mais vous pouvez utiliser tout autre canal.
mixerbar	sortie	affiche une barre d'état du volume de 'sortie'.

mixerl	sortie	affiche le volume gauche de 'sortie'.
mixerlbar	sortie	affiche une barre d'état du volume gauche de 'sortie'.
mixerr	sortie	affiche le volume droit de 'sortie'.
mixerrbar	sortie	affiche une barre d'état du volume droit de 'sortie'.
section moc* - ces variables renvoient les informations délivrées par MusicOnConsole..		
moc_album	-	affiche l'album courrant en lecture dans moc.
moc_artist	-	affiche l'artiste en lecture dans moc.
moc_bitrate	-	affiche le debit du morceau en lecture dans moc.
moc_curtime	-	affiche le temps courant du morceau en lecture dans moc.
moc_file	-	affiche le nom complet du fichier en lecture dans moc.
moc_rate	-	affiche le classement du morceau en lecture dans moc.
moc_song	-	affiche le nom du morceau en lecture dans moc.
moc_state	-	affiche l'état de moc, 'en lecture' 'en pause' 'stoppé' ...
moc_timeleft	-	affiche le temps restant dans le morceau en lecture dans moc.
moc_title	-	affiche le titre du morceau en lecture dans moc.
moc_totaltime	-	affiche la durée totale du morceau en lecture dans moc.
fin de la section moc.		
monitor	-	affice le numéro du moniteur sur lequel tourne conky.
monitor_number	-	affiche le nombre de moniteurs.
section mpd* - ces variables renvoient les informations délivrées par MusicPlayerDaemon .		
mpd_album	-	affiche l'album en lecture dans mpd.
mpd_artist	-	affiche l'artiste en lecture dans mpd.
mpd_bar	hauteur,longueur	affiche une barre d'état du morceau en lecture dans mpd.
mpd_bitrate	-	affiche le debit du morceau en lecture dans mpd.
mpd_elapsed	-	affiche le temps restant de la lecture en cours dans mpd.
mpd_file	-	affiche le nom complet du fichier en lecture dans mpd.
mpd_length	-	affiche la durée du morceau en lecture dans mpd.
mpd_name	-	affiche le nom du morceau en lecture dans mpd.
mpd_percent	-	affiche le pourcentage de progression du morceau en lecture dans mpd.
mpd_random	-	affiche le statut du mode aléatoire.
mpd_repeat	-	affiche le statut du mode répétition.
mpd_smart	nombre 'n'	affiche le nom, sous la forme 'artiste - titre' selon les informations disponibles. taille maxi='n'.
mpd_status	-	affiche le statut de mpd.
mpd_title	nombre 'n'	affiche le titre du morceau en lecture dans mpd. taille maxi='n'
mpd_track	-	affiche le numéro du morceau en lecture dans mpd.
mpd_vol	-	affiche le volume de mpd.
fin de la section mpd* .		
nameserver	index	affiche les informations de 'nameserver', voir /etc/resolv.conf.
new_mails	boîte_mail	affiche le nombre de messages non-lus dans votre

	intervalle	'boîte_mail', vérifié tous le 'intervalles' en minutes.
nodename	-	affiche le nom d'hôte du système.
nvidia	threshold temp ambient gpufreq memfreq imagequality	affiche les informations relative à votre carte grapique nvidia™.
offset	nombre (pixels)	déplace le prochain élément de 'n' pixels.
outlinecolor	hexadécimal/couleur	change la couleur du contour de la police.
pb_battery	status,percent,time	affiche des informations sur la batterie des pwerbook/ibook™.
section pid* - ces variables renvoient des informations sur les processus en fonction de leur PID.		
pid_chroot	pid	affiche l'environnement de travail de 'pid'
pid_cmdline	pid	affiche la ligne de commande dont 'pid' est issu.
pid_cwd	pid	affiche le répertoire de travail de 'pid'.
pid_environ	pid varname	Fix Me!
pid_environ_list	pid	Fix Me!
pid_exe	pid	affiche l'adresse de l'executable à l'origine de 'pid'.
pid_nice	pid	affiche le 'nice' de 'pid'.
pid_openfiles	pid	affiche une liste des fichiers ouverts par 'pid'.
pid_parent	pid	affiche le pid du processus parent de 'pid'.
pid_priority	pid	affiche la priorité de 'pid', voir 'priority' dans 'man 5 proc'.
pid_read	pid	affiche le nombre de bytes lus par 'pid'.
pid_state	pid	affiche le statut de 'pid', 'actif' 'au repos' 'zombie'...
pid_state_short	pid	affiche le statut de 'pid' en format court.
pid_stderr	pid	affiche les erreurs liées au 'pid'.
pid_stdin	pid	affiche les données entrantes au 'pid'.
pid_stdout	pid	affiche les données sortantes du 'pid'.
pid_threads	pid	affiche le nombre de tâches comprises dans 'pid'.
pid_thread_list	pid	affiche une liste des tâches de 'pid'.
pid_time_kernelmode	pid	affiche le temps en secondes alloué par le noyau pour 'pid'
pid_time_usermode	pid	affiche le temps en secondes alloué par l'utilisateur pour 'pid'
pid_time	pid	affiche la somme de 'pid_time_kernelmode'+'pid_time_usermode' pour 'pid'.
pid_uid	pid	affiche le 'uid' réel pour 'pid'.
pid_euid	pid	affiche le 'uid' effectif pour 'pid'.
pid_suid	pid	affiche le 'uid' sauvegardé pour'pid'.
pid_fsuid	pid	affiche le 'uid' du système de fichiers pour 'pid'.
pid_gid	pid	affiche le 'gid' réel pour 'pid'.
pid_egid	pid	affiche le 'gid' effectif pour 'pid'.
pid_sgid	pid	affiche le 'gid' sauvegardé pour 'pid'.
pid_fsgid	pid	affiche le 'gid“ du système de fichiers pour 'pid'.

pid_vmpeak	pid	affiche le pic de mémoire virtuelle utilisée par 'pid'.
pid_vmsize	pid	affiche la quantité de mémoire utilisée par 'pid'.
pid_vmlck	pid	affiche la quantité de mémoire réservée pour 'pid'.
pid_vmhwm	pid	Fix Me!
pid_vmrss	pid	Fix Me!
pid_vmdata	pid	Fix Me!
pid_vmstk	pid	Fix Me!
pid_vmexe	pid	Fix Me!
pid_vmlib	pid	Fix Me!
pid_vmpte	pid	Fix Me!
pid_write	pid	affiche le nombre de bytes écris par 'pid'
fin de la section pid*		
platform	dev type n factor offset	affiche les informations transmises par la sonde platform de sysfs (Linux2.6). 'dev' peut être ignoré si vous n'avez qu'un seul capteur. le type peut être 'in'/'vol' pour voltage, 'fan' ou 'temp'. 'n' est le numéro de la sonde (voir /sys/bus/platform/devices/). 'factor' et 'offset' sont optionnels et autorisent le recalcul des données entrantes.
pop3_unseen	affiche le nombre de message dans votre boîte mail POP3 (voir pop3). vous pouvez définir des boîtes pop3 individuelles avec les arguments suivants: “host user pass [-i interval] [-p port] [-e command]”. Port par défaut : 110, intervalle par défaut : 5 minutes. Si le mot de passe est '*', il vous sera demander au démarrage de Conky.	
pop3_used	affiche l'espace utilisé dans votre boîte mail POP3 (voir pop3). vous pouvez définir des boîtes pop3 individuelles avec les arguments suivants: “host user pass [-i interval] [-p port] [-e command]”. Port par défaut : 110, intervalle par défaut : 5 minutes. Si le mot de passe est '*', il vous sera demander au démarrage de Conky.	
pre_exec	shell commande	execute une commande une fois avant que conky n'affiche quoi que ce soit, puis affiche le résultat dans la zone TEXT.
processes	-	nombre total de processus (actif/au repos).
read_tcp	hôte port	affiche les caractères visibles en se connectant via 'hôte' sur 'port' tcp.
replied_mails	maildir intervalle	nombre de mails auxquels vous avez répondus dans votre 'maildir'.
rss	adresse intervalle action nombre'n' space_in_front	télécharge les flux rss à un 'intervalle' déterminé (par défaut 15 minutes), et renvoi les 'actions': feed_title, item_title('n'), item_desc('n'), 'item_titles'. possibilité d'utiliser 'spaces_in_front' pour ajouter un espace avant chaque 'action'. ex(archwiki): '${rss http://planet.archlinux.org/rss20.xml 1 item_titles 10 }', affichera les dix premiers articles de archlinux planet.
running_processes	-	affiche le nombre de processus actifs.
running_threads	-	affiche le nombre de tâches en cours.
scroll	longueur('l') step text	affiche un 'text' défilant de 'l' caractères avec un défilement de 'step' caractères.

		'text' peut contenir des variables, si une variable renvoi plusieurs lignes, elles seront placées à la suite avec un "|" de séparation. si vous changer la couleur , elle sera restaurée à la fin de '$scroll'. la fin et le début de 'text' sont séparés par 'l' espaces.
seen_mails	maildir intervalle	affiche le nombre de mails marqués comme “lus” dans votre 'maildir'
shadecolor	hexadécimal/couleur	change la couleur de l'ombre des textes/graphs/bordures.
section smapi - voir /sys/devices/platform/smapi/ - préférer les commandes “smapi_*” .		
smapi (obsolète, préférer smapi_*)		
smapi_bat_bar	index,hauteur,longueur	affiche une bare d'état de la batterie.
smapi_bat_perc	index,hauteur,longueur	idem que 'smapi_bat_bar' mais accepte la fonction 'user_spacer'.
smapi_bat_power	index	affiche la puissance en watt, posotif en charge, négatif en décharge.
smapi_bat_temp	index	affiche la température de la batterie.
fin de la section smapi*.		
sony_fanspeed	-	affiche la vitesse de rotation des ordinateurs portables SonyVAIO™.
stippled_hr	'n' pixels	trace une ligne horizontale en pointillés d'épaisseur 'n'.
swap	-	affiche l'utilastion de l'espace d'échange 'swap'.
swapbar	hauteur,longueur	affiche une barre d'état de l'utilisation de la swap.
swapfree	-	affiche la place disponible dans la swap.
swapmax	-	affiche la taille totale de la swap.
swapperc	-	affiche le pourcentage d'utilisation de la swap.
sysname	-	affiche le nom du système, ex: Linux.
tab	longueur'l',debut	insère une tabulation de 'l' pixels commencant à la colonne 'début'.
tail	fichier 'n'lignes check	affiche les 'n' dernières lignes du 'fichier' vérifié tous les 'check'.
tcp_portmon	moniteur de ports TCP. Les numéros de port doivent doivent être dans l'intervalle 1 à 65535. Les arguments possibles sont: 'count' nombre total de connexions dans l'intervalle spécifié, 'rip' adresse ip distante, 'rhost' nom de l'hôte distant, 'rport' numéro du port distant, 'rservice' nom du service distant (depuis /etc/services), 'lip' adresse ip locale, 'lhost' nom de l'hôte local, 'lport' numéro du port local, 'lservice' nom du service local (depuis /etc/services). L'index de connexion fournit l'accès à chaque connexion au moniteur de port. Celui-ci vous retourne les informations pour des valeurs d'index comprises entre les connexions 0 et n-1. Les valeurs au-dessus de n-1 sont ignorées. Pour l'argument 'count', l'index de connection ne doit pas être spécifié mais est requis pour tous les autres arguments. Examples: '${tcp_portmon 6881 6999 count}' affiche le nombre de connexions dans la plage d'ip bittorrent. '${tcp_portmon 22 22 rip 0}' affiche l'adresse ip distante de la première connexion ssh	

	'${tcp_portmon 22 22 rip 9}' affiche l'adresse ip distante de la dixième connexion ssh '${tcp_portmon 1 1024 rhost 0}' affiche le nom d'hôte distant de la première connexion sur le port indiqué (1024 ici) '${tcp_portmon 1 1024 rport 4}' affiche le port distant de la cinquième connexion sur le port indiqué '${tcp_portmon 1 65535 lservice 14}' affiche le nom du service local de la quinzième connexion dans la plage complète (1 à 65535) NB: les variables du moniteur de port qui partage le même port font référence au même moniteur, ainsi plusieurs références à une plage particulière pour différentes infos et différents index utilisent en interne le même moniteur. Autrement dit Conky ne crée pas de moniteurs redondants.	
templateN	Fix Me!	Fix Me!
texeci	intervalle commande	exécute 'commande' toutes les 'intervalle' secondes et affiche la sortie. idem que $execi, excepté que la commande est exécutée dans une tâche porpre. utilisez cette variable si vous avez un script assez lent afin que Conky continue le rafraîchissement. mettez un intervalle plus long que le temps d'éxécution que votre script.
threads	-	affiche le nombre total de tâches.
time	format	affiche la date, l'heure... voir 'man stfrtime'
to_bytes	nombre	si 'nombre' est suivi d'une unités de meure(KiB,MB,GiB..), il est convertit en bytes sans unités.
top	type numéro	classement (de 1→10) des processus selon leur utilisation processeur. les 'types' sont: 'name','pid','cpu','mem','mem_res','mem_vsize','time','io_perc','io_read','io_write'.
top_io	type numéro	idem que 'top' mais les processus sont classés par la quantité de données en lecture/écriture pendant l'intervalle de lecture.
top_mem	type numéro	idem que 'top' mais les processus sont classés par leur taux d'occupation mémoire.
top_time	type numéro	idem que 'top' mais les processus sont classés par leur temps cpu.
totaldown	interface	affiche la quantité totale de données téléchargées (limité à 4GB).
trashed_mails	maildir intervalle	affiche le nombre de mails dans la poubelle de 'maildir'.
tztime	timezone format	idem que 'time' mais pour un fuseau horraire spécifique. voir /usr/share/zoneinfo.
gid_name	gid	affiche le groupe correspondant à 'gid'.
uid_name	uid	affiche le nom de l'utisateur 'uid'.
unflagged_mails	maildir intervalle	affiche le nombre de mails non classée dans 'maildir'.
unforwarded_mails	maildir intervalle	affiche le nombre de mails classés non transférés dans 'maildir'.
unreplied_mails	maildir intervalle	affiche le nombre de mails classés sans réponses dans 'maildir'.
unseen_mails	maildir intervalle	affiche le nombre de nouveaux mails dans 'maildir'.
updates	*nombre*	*pour le débug*.

	d'updates	
upspeed	interface	affiche la quantité de données téléversées pour 'interface'.
upspeedf	interface	affiche la quantité de données téléversées en KiB avec une décimale pour 'interface'.
upspeedgraph	interface hauteur,longueur couleur1 couleur2 scale -t -l	affiche un graphique des données téléversées pour 'interface', ex: '${upspeedgraph wlan0 10,60 FFFFFF 000000}'.
uptime	-	affiche le temps de connexion.
uptime_short	-	affiche le temps de connexion en format court.
user_names	-	affiche le nom des utilisateurs connectés.
user_number	-	affiche le nombre d'utilisateurs connectés.
user_terms	-	affiche une liste des consoles en utilisation.
user_times	-	affiche les temps de connexion des utilisateurs.
user_time	console	affiche le temps de connexion de l'utilisateur connecté en 'console'.
utime	format	affiche l'heure UTC.
voffset	nombre'n' (pixels)	décallage vertical de 'n' pixels pour l'élément suivant.
voltage_mv	'n'	affiche la consommation du cpu 'n' en mVolt.
voltage_v	'n'	affiche la consommation du cpu 'n' en Volt.
section “weather” - “weather_forecast” - préférer les scripts externes pour la météo.		
weather	Fix Me!	Fix Me!
weather_forecast	Fix Me!	Fix Me!
fin de la secton météo.		
section wireless - ces variables renvoient des informations à propos de votre réseau sans-fil.		
wireless_ap	interface	affiche le point d'accès MAC du réseau.
wireless_bitrate	interface	affiche le débit du réseau sans-fil.
wireless_essid	interface	affiche le point d'accès ESSID du réseau.
wireless_link_bar	hauteur,longueur interface	affiche une barre d'état de la qualité du réseau.
wireless_link_qual	interface	affiche la qualité du signal sans-fil.
wireless_link_qual_max	interface	affiche la qualité maximale du signal disponible sur le réseau.
wireless_link_qual_perc	interface	affiche la qualité du signal en pourcentage.
wireless_mode	interface	affiche le mode du réseau (Managed/Ad-Hoc/Master).
fin de la secton wireless.		
words	fichier	affiche le nombre de mots dans le 'fichier'.
section xmms2 - ces variables renvoient les informations délivrées par le lecteur xmms2.		
xmms2_album	-	affiche l'album du morceau en lecture dans xmms2.
xmms2_artist	-	affiche l'artiste du morceau en lecture dans xmms2.
xmms2_bar	hauteur,longueur	affiche une barre de progression de la lecture en cours dans xmms2.
xmms2_bitrate	-	affiche le débit du morceau en lecture dans xmms2.
xmms2_comment	-	affiche les commentaires du morceau en lecture dans

		xmms2.
xmms2_date	-	affiche la date du morceau en lecture dans xmms2.
xmms2_duration	-	affiche la durée totale du morceau en lecture dans xmms2.
xmms2_elapsed	-	affiche le temps restant du morceau en lecture dans xmms2.
xmms2_genre	-	affiche le genre du morceau en lecture dans xmms2.
xmms2_id	-	affiche l'id du morceau en lecture dans xmms2.
xmms2_percent	-	affiche le pourcentage de lecture du morceau en cours dans xmms2.
xmms2_playlist	-	affiche la playlist en cours dans xmms2.
xmms2_size	-	affiche la taille du morceau en cours dans xmms2.
xmms2_smart	-	affiche le morceau en cours dans xmms2 sous le format 'artise - titre' selon les informations disponibles.
xmms2_status	-	affiche le statut de xmms2.
xmms2_timesplayed	-	affiche le nombre de lecture du morceau en cours dans xmms2.
xmms2_title	-	affiche le titre du morceau en lecture dans xmms2.
xmms2_tracknr	-	affiche le numéro du morceau en lecture dans xmms2.
xmms2_url	-	affiche l'adrese internet du morceau en lecture dans xmms2.
fin de la section xmms2.		

contributeur: arpinux