les menus indépendants

étant un grand fan de minimalisme, et donc pas vraiment attiré vers les environnements de bureaux complets (Gnome/KDE/E17) je me trouve souvent confronté au problème du menu. bien sûr les raccourcis clavier sont parfaits, mais que voulez-vous, j'aime avoir un menu...
voici quelques uns des menus indépendants que j'utilise: associés à un raccourcis clavier, un lanceur dans un panel/dock ou intégré dans une zone de notification.

Sommaire

compiz-deskmenu

compiz-deskmenu est un menu construit à la base pour compiz, vous l'aurez compris, mais il peut s'intégrer à tous les gestionnaires de fenêtres pour obtenir un menu principal ou secondaire, lancé depuis un panel/lanceur, associé à un raccourcis clavier, ou à une action de la souris sur le bureau (clic droit comme le menu openbox).

installation

paquet Debian

une ptit paquet deb pour faire vite ? ⇒ http://arpinux.org/public/pkgs/compiz-deskmenu-alldeb.deb

depuis les sources

compiz-deskmenu est disponible dans les dépôts AUR pour les utilisateurs de ArchLinux. pour les débianistes, qui installent depuis les sources, on commence par les dépendances :

```
# apt-get install python-lxml python-xdg libgtk2.0-dev libwnck-dev libdbus-1-dev
git-core libdbus-glib-1-dev
```

puis on va chercher les sources sur git

```
$ git clone git://anongit.compiz.org/users/crdlb/compiz-deskmenu
```

et enfin on installe

```
$ cd compiz-deskmenu
$ make
# make install
```

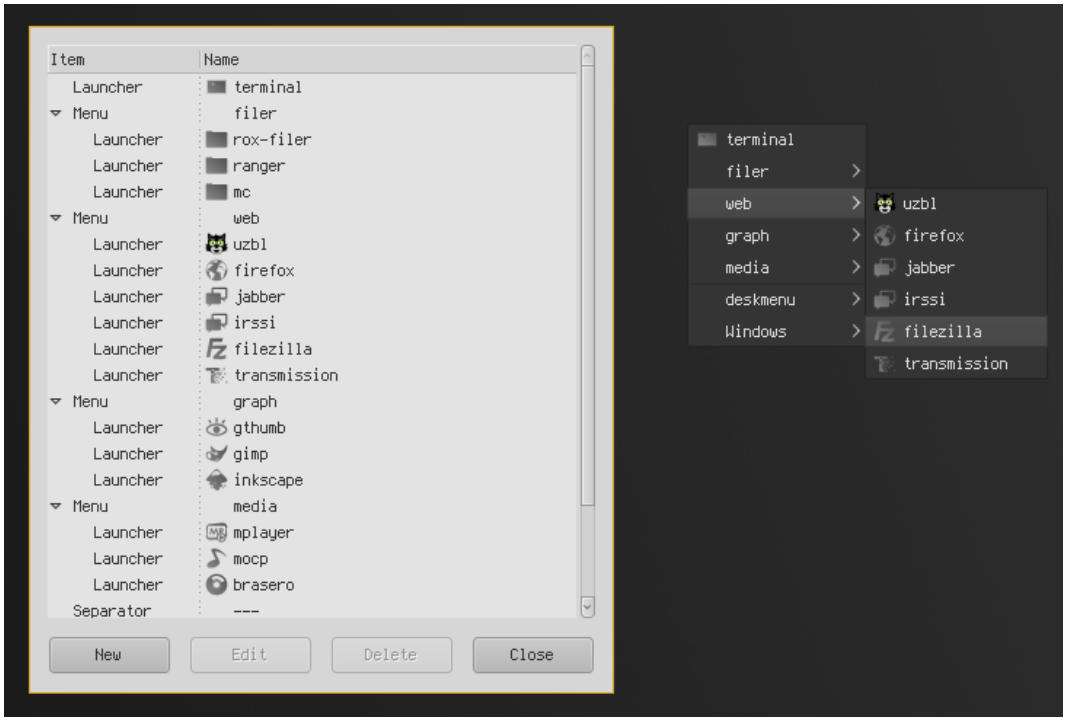
pour lancer compiz-deskmenu

```
$ compiz-deskmenu &
```

configuration

compiz-deskmenu dispose de son propre outil de configuration **compiz-deskmenu-editor** accessible depuis le menu lui-même ou depuis un terminal

```
$ compiz-deskmenu-editor
```



cet outils va modifier le fichier de configuration de compiz-deskmenu situé dans votre ~/.config/compiz/deskmenu/menu.xml. voici celui utilisé pour l'exemple du dessus:

```
<menu>
  <item type="launcher">
    <name>terminal</name>
    <icon>gnome-terminal</icon>
    <command>urxvtc</command>
  </item>
  <menu name="filer">
    <item type="launcher">
      <name>rox-filer</name>
      <icon>nautilus</icon>
      <command>rox-filer</command>
    </item>
    <item type="launcher">
      <name>ranger</name>
      <command>urxvtc -e ranger</command>
      <icon>nautilus</icon>
    </item>
    <item type="launcher">
      <name>mc</name>
      <command>urxvtc -e mc</command>
      <icon>nautilus</icon>
    </item>
  </menu>
  <menu name="web">
    <item type="launcher">
      <name>uzbl</name>
      <icon>uzbl</icon>
      <command>uzbl http://arpinux.org</command>
    </item>
    <item type="launcher">
      <name>firefox</name>
      <icon>gnome-web-browser</icon>
      <command>firefox</command>
```

```
    </item>
    <item type="launcher">
      <name>jabber</name>
      <icon>pidgin</icon>
      <command>urxvtc -g 114x44+1156+84 -e mcabber</command>
    </item>
    <item type="launcher">
      <name>irssi</name>
      <icon>pidgin</icon>
      <command>urxvtc -g 114x44+1156+84 -e irssi</command>
    </item>
    <item type="launcher">
      <name>filezilla</name>
      <icon>filezilla</icon>
      <command>filezilla</command>
    </item>
    <item type="launcher">
      <name>transmission</name>
      <icon>transmission</icon>
      <command>transmission</command>
    </item>
  </menu>
  <menu name="graph">
    <item type="launcher">
      <name>gthumb</name>
      <icon>gthumb</icon>
      <command>gthumb</command>
    </item>
    <item type="launcher">
      <name>gimp</name>
      <icon>gimp</icon>
      <command>gimp</command>
    </item>
    <item type="launcher">
      <name>inkscape</name>
      <icon>inkscape</icon>
      <command>inkscape</command>
    </item>
  </menu>
  <menu name="media">
    <item type="launcher">
      <name>mplayer</name>
      <icon>mplayer</icon>
      <command>gnome-mplayer</command>
    </item>
    <item type="launcher">
      <name>mocp</name>
      <icon>rhythmbox</icon>
      <command>urxvtc -e mocp -T transparent-background</command>
    </item>
    <item type="launcher">
      <name>brasero</name>
      <icon>brasero</icon>
      <command>brasero</command>
    </item>
  </menu>
  <separator/>
  <menu name="deskmenu">
```

```
    <item type="launcher">
      <icon>gtk-edit</icon>
      <command>compiz-deskmenu-editor</command>
      <name>edit menu</name>
    </item>
    <item type="reload"/>
  </menu>
  <item type="windowlist"/>
</menu>
```

comme vous le voyez, le menu xml accepte les commandes avec arguments. mais l'outils graphique intégré fonctionne très bien, alors à moins d'être un accro à vim, vous ne l'éditerez certainement pas à la main :).

MyGTKMenu

sur la page officielle du site , MyGTKMenu est présenté comme un script qui lit un texte ... et c'est exactement ce que c'est :) la dernière version est la 1.3 et demande GTK-2.4 pour fonctionner. il s'intègre à tous les environements, adopte le thème gtk et supporte les icones.

installation

il suffit de télécharger l'archive sur le site principal ou sur mon serveur et de décompresser l'archive dans un dossier. vous disposez des ces fichiers/dossiers

- myGtkMenu : un executable 32bits utilisant GTK+-2.x
- TestMenu.txt : un exemple de configuration de menu
- main.c : le fichier en C pour pouvoir recompiler en cas d'utilisation dur 64bits
- Makefile: le makefile pour 64bits
- License.txt
- README : même si c'est en anglais, lisez-le ...
- gnome-icons : les icones appellées par myGtkMenu
- icons : des icones de myGtkMenu pour lanceur

placez myGtkMenu dans un dossier d'executables, assurez vous qu'il le soit, puis lancer la commande

```
myGtkMenu <chemin vers TestMenu.txt>
```

configuration

ici pas d'outils intégré, il faut éditer le fichier de menu à la main. l'intérêt est que vous pouvez créer autant de menus personnalisés que vous désirez et les lancez avec un simple

```
myGtkMenu <chemin vers Menu1>
myGtkMenu <chemin vers Menu2>
..
```

par exemple, un menu internet avec vos liens favoris

```
#      ............... Beginning of menu ..............

#menupos = 10 10 # Optional

iconsize = 25

item = Liens favoris
cmd = " "
icon = <chemin vers mon icone qui tape>

separator

item =  arpinux.org
cmd = firefox http://arpinux.org
icon = <chemin vers mon autre icone qui tape>

separator

submenu = shopping

     icon = <chemin vers mon icone shopping>

     item = chaussures
     cmd = firefox http://www.lachaussurepascher.fr
     icon = <chemin vers mon icone de chaussures>

     item = voitures
     cmd = firefox http://vaspassertonpermis.org
     icon = <chemin vers mon icone de voiture>

separator
..
..
#### end of menu ####
```

voici la liste des arguments accepté dans le fichier de menu:

- menupos : position du menu sur l'écran. optionnel
- item : texte affiché
- cmd : commande à executer
- icon : chemin complet vers l'icone sinon utiliser “icon = NULL”
- separator : séparateur
- submenu : créer un sous-menu
- iconsize : determine la taille de l'icone

notez que l'identation est importante dans ce fichier: un sous-menu doit obligatoirement marquer un décallage avec le menu parent.

vous pouvez désormais profiter de votre (vos) menus depuis un lanceur, un raccourcis ou l'associer au clic-droit sur votre bureau.

9Menu

9menu est un menu ultra minimal qui se lance depuis la ligne de commande ou lit un fichier de configuration. il est très léger et accepte une syntaxe simple.

installation

9menu est présent dans les dépots de la plupart des distributions. pour les debianistes:

```
# apt-get install 9menu
```

syntaxe (issue du man):

```
9menu  [  -bg background-color ] [ -display displayname ] [ -file name ] [ -fg
foreground-color ] [ -font fname ] [ -geometry geom ] [ -iconic ] [ -label name ]
[ -path ] [ -popdown ] [ -popup ] [ -shell prog ] [ -teleport ] [ -version ] [
-warp  ]  menuitem[:command]
```

- bg: couleur de fond
- display: écran à utiliser
- file: nom de fichier à utiliser
- fg: couleur du texte
- font: police utilisée
- geometry: taille et placement du menu
- iconic: démarrage iconifié
- label: nom de la fenêtre de menu
- path: Fix Me!
- popdown: lorsqu'une entrée est slectionné, le menu s'iconifie
- popup: lorsqu'une entrée est slectionné, le menu quitte
- shell: définir un autre shell que /bin/sh
- teleport: place le menu sous la fenêtre
- version: affiche la version
- warp: après sélection, place la souris dans sa position initiale

configuration

nous allons prendre deux exemples: un menu basique à simple entrée, et un menu évolué avec des sous-menus.

menu simple

dans notre exemple, nous utiliserons 9menu directement depuis la commande passée depuis un script:

```
#!/bin/sh
# file: ~/bin/echimenu.sh
# lanceur de menu simple
exec 9menu -bg grey30 -fg grey80 -font -*-terminus-medium-*-*-*-12-*-*-*-*-*-*-*
-popup -label 'menu' 'terminal:urxvtc' 'rox-filer:rox' 'ranger:xterm -title
ranger -e ranger' 'uzbl:uzbl' 'firefox:firefox' '@jabber:xterm -title jabber -e
mcabber' 'editor:xterm -title editor -e vim' '' 'xcompmgr:xcompmgr_reset'
'lock:slock'  '! halt !:sudo halt'
```

ce script, que vous prendrez soin de rendre éxécutable, lance un menu ultra minimal qui ressemble à ceci:

menu avec sous-menus

9menu ne supporte pas les sous-menus par défaut mais une astuce permet de lancer d'autres instances de 9menu depuis une instance principale: ce qui permet la création de sous-menus. (*sources*: l'excellent post de thuban sur son blog.)
pour plus de facilité dans la gestion des fichiers des menus, je vous invite à la création d'un dossier regroupant votre menu principal et ses sous-menus:

```
$ mkdir ~/.9menu
$ cd ~/.9menu/
$ touch main_menu && touch internet && touch graphisme && touch office && touch
multimedia && touch outils && touch system
```

il nous faut éditer le script de lancement principal que vous prendrez soin de placer dans un dossier d'éxécutables (inclus dans $PATH):

```
#!/bin/sh
# file: ~/bin/main_menu.sh
# auteur: thuban <http://thuban.toile-libre.org/>
# 9menu-full-script
#Apparence
BG="#08090A"
FG="#4B555E"
FN="-*-terminus-medium-*-*-*-12-*-*-*-*-*-*-*"
# Le dossier du menu
DIR=~/.9menu/
# Le fichier du menu
FILE=main_menu
cd $DIR
9menu -label "full-menu" -popup -bg "$BG" -fg "$FG" -font "$FN" -teleport -file
$FILE
```

ce script lance le “main_menu” situé dans votre ~/.9menu/:

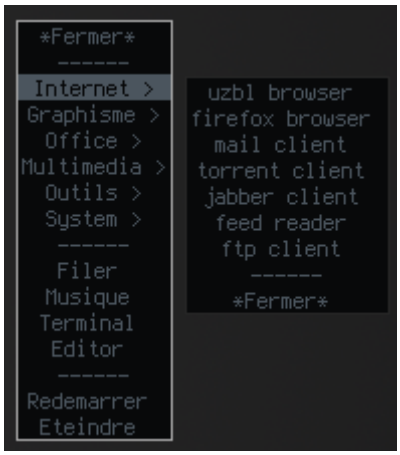
```
*Fermer* :exit
\------
Internet >:9menu -label 'internet' -popup -bg '#08090A' -fg '#4B555E' -font '-*-
terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file internet
Graphisme >:9menu -label 'graphisme' -popup -bg '#08090A' -fg '#4B555E' -font '-
```

```
*-terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file graphisme
Office >:9menu -label 'office' -popup -bg '#08090A' -fg '#4B555E' -font '-*-
terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file office
Multimedia >:9menu -label 'multimedia' -popup -bg '#08090A' -fg '#4B555E' -font
'-*-terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file multimedia
Outils >:9menu -label 'outils' -popup -bg '#08090A' -fg '#4B555E' -font '-*-
terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file outils
System >:9menu -label 'system' -popup -bg '#08090A' -fg '#4B555E' -font '-*-
terminus-medium-*-*-*-12-*-*-*-*-*-*-*' -teleport -file system
\------
Filer :cd ~ && rox-filer
Musique :urxvtc -title player -e mocp -T transparent-background
Terminal :cd ~ && urxvtc
Editor :urxvtc -e vim
\------
Redemarrer :sudo shutdown -r now
Eteindre :sudo shutdown -h now
```

chaque entrée de sous-menu lance une autre instance de 9menu qui lit un fichier de config situé dans votre ~/.9menu. un exemple avec le sous-menu 'internet' qui renvoi au fichier ~/.9menu/internet:

```
uzbl browser :uzbl
firefox browser :firefox
mail client :claws-mail
torrent client :transmission
jabber client :urxvtc -title jabber -e mcabber
feed reader :urxvtc -title feeds -e canto
ftp client :filezilla
\------
*Fermer* :exit
```

il ne vous reste plus qu'à éditer vos autres sous-menus et vous obtiendrez un menu ressemblant à ce résultat:



ratmenu

ratmenu est un fork de 9menu pilotable uniquement au clavier conçu pour ratpoison.

installation

ratmenu est disponible sur la plupart des distributions GNU/Linux et s'utilise sur tous les wms.

pour Debian:

```
# apt-get install ratmenu
```

configuration

la configuration passe par la ligne de commande. pour l'exemple, mon ratmenu (utilisé sur dwm)

et le script qui le lance:

```
#!/bin/sh
exec ratmenu -style "snazzy" -label "rmenu" -align "center" -fg "#BFBFBF" -bg
"#005885" -font "-artwiz-lime-medium-*-normal-*-10-*-*-*-*-*-*-*" \
    "claws-mail" "claws-mail --online" \
    "conkeror" "conkeror" \
    "firewall" "urxvtcd -e sudo vim /etc/firewall.rules" \
    "gcolor2" "gcolor2" \
    "geany" "geany" \
    "gimp" "gimp" \
    "gparted" "gksudo gparted" \
    "radio-cli" "urxvtcd -e cli_radio" \
    "ranger" "urxvtcd -e ranger" \
    "rox-filer" "rox" \
    "synaptic" "gksudo synaptic" \
    "torrent" "transmission" \
    "unetbootin" "gksudo unetbootin" \
    "unison" "sudo unison-gtk" \
    "volume" "urxvtcd -title sound -e alsamixer" \
    "xosview" "xosview" \
    "xcolorsel" "xcolorsel" \
    "xfontsel" "xfontsel" \
```

```
    "----" ""\
    "xinit" "urxvtcd -e vim .xinitrc" \
    "start" "urxvtcd -e vim bin/dwm-session" \
    "xdefs" "urxvtcd -e vim .Xresources" \
    "firewall" "urxvtcd -e sudo vim /etc/firewall.rules" \
    "----" ""\
    "lock" "slock" \
    "exit" "sudo halt"
exit 0
```

les options acceptées

les options peuvent être fixées dans votre ~/.Xresources. notez que les options passées en ligne de commande écrasent celles du ~/.Xresources.

option	argument	description
-display	displayname	utilise l'écran indiqué au lieu de l'écran par défaut.
-font	fname	(X Resource: font) utilise la police spécifiée au lieu de celle par défaut.
-label	name	change le nom de la fenêtre de menu et de l'icône. le nom par défaut est la dernière commande passée àa ratmenu lors du lancement, généralement, ratmenu..
-fg	foreground-color	(X Resource: fgcolor) couleur du texte, par défaut: black.
-bg	background-color	(X Resource: bgcolor) couleur du fond, par défaut: white.
-io	item-offset	item-offset deviant la première entrée de ratmenu. par défaut, la première entrée correspond à la première commande , ou item-offset 1. la deuxième commande sera item-offset 2.
-style	{snazzy:dreary}	(X Resource: style) snazzy est le style par défaut, la barre de sélection reste fixe et les éléments du menus défilent. en mode dreary, la barre de sélection monte et descend sur le menu.
-align	{left:center:right}	(X Resource: align) alignement des entrées de menu. left par défaut.
-shell	prog	utilise 'prog' à la place de /bin/sh. si le prog n'est pas trouvé, ratemenu retourne sur /bin/sh.
-back	prevmenu	commande à lancer quand la commande 'back' est lancé, généralement pour créer un ratmenu à plusieurs niveaux.
-persist		(X Resource: persist) empâche ratmenu de se fermer lors d'une sélection.
-version		affiche la version puis exit0.

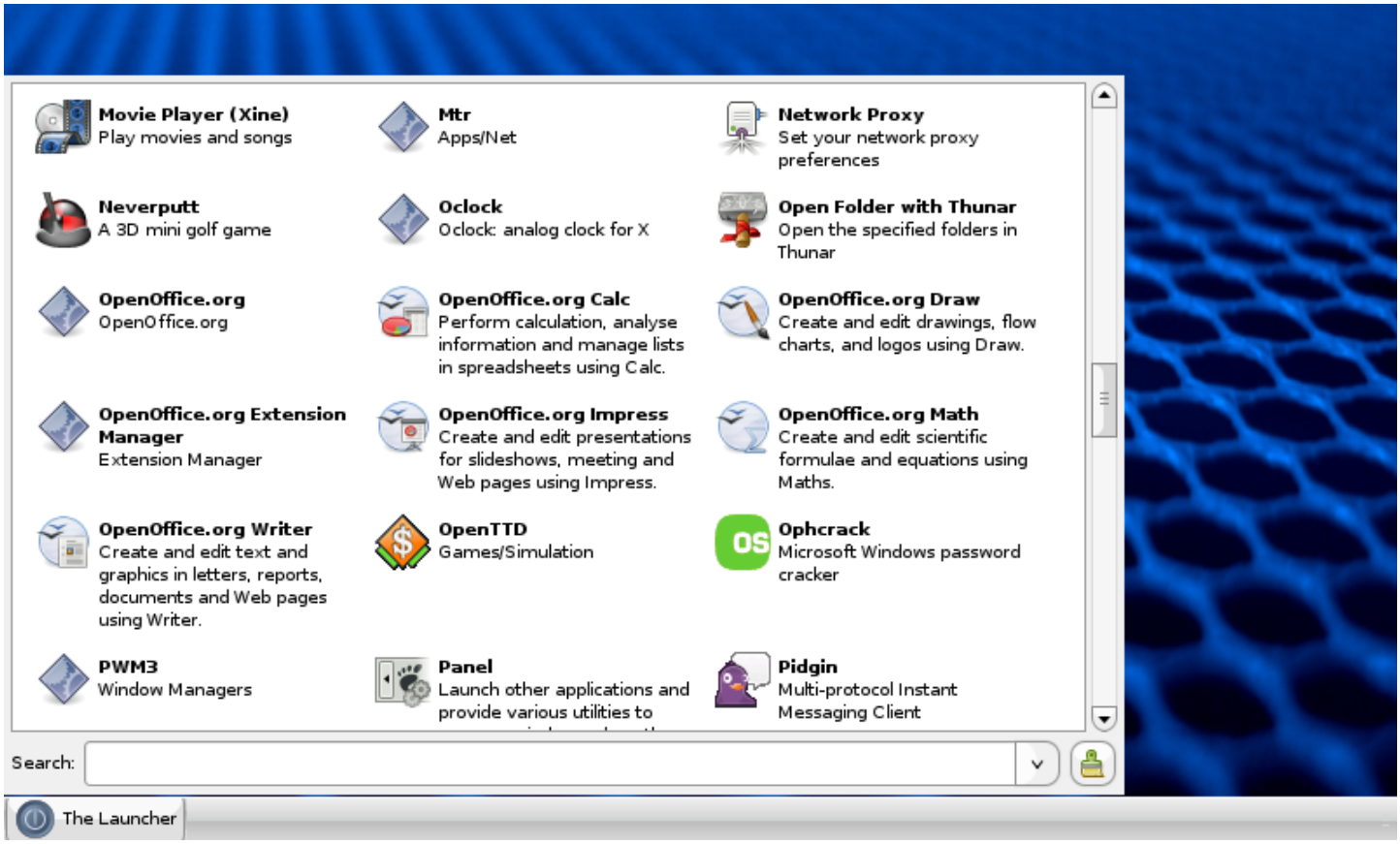
theLauncher

et voici une petite application en python développée par Alexander Barinov, initialament prévu pour être un applet de gnome-panel. mais il peut de lancer en stand-alone et dispose même d'un lanceur sous la forme d'un applet pour la zone de notification. il fonctionne de façon simple: theLauncher liste les applications contenues dans votre /usr/share/applications puis en dresse un menu ordonné s'affichant dans une fenêtre.

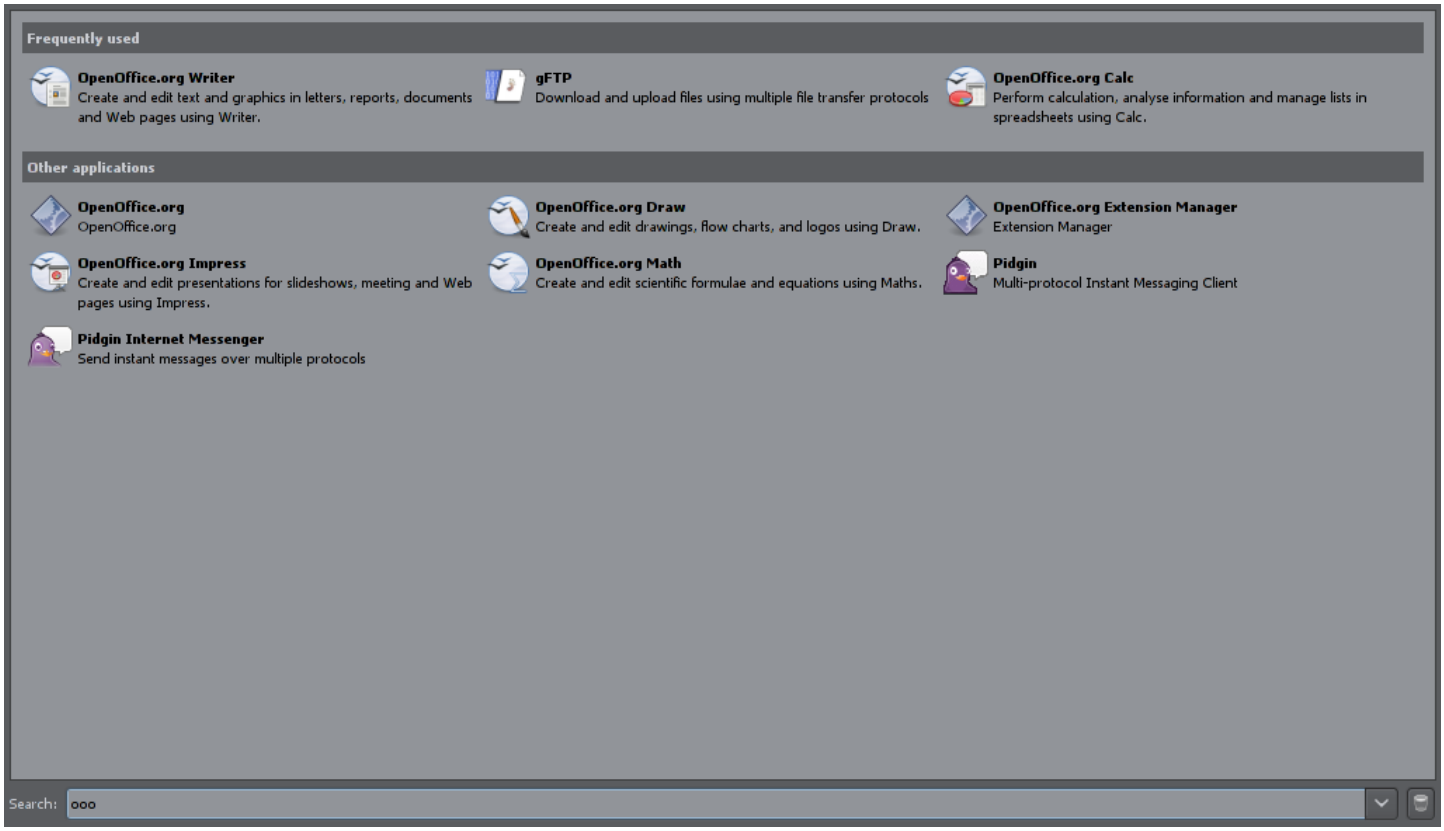
screenshots

thelauncher en stand-alone, depuis gnome-panel, en mode recherche, en mode nouvelle application...

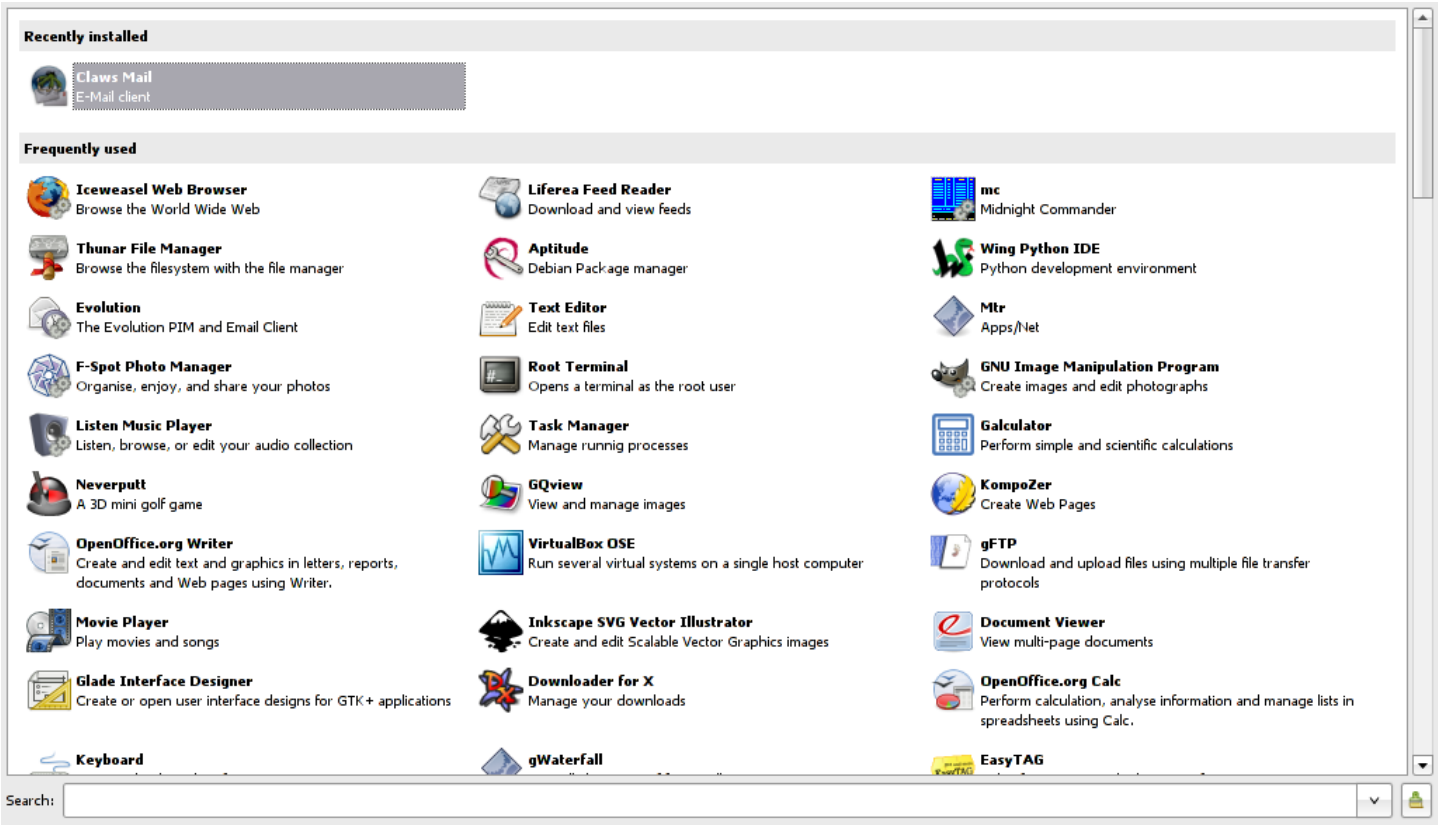
thelauncher en stand-alone



thelauncher depuis gnome-panel



thelauncher recherche



thelauncher nouvelle application

installation & utilisation

thelauncher dépend de **python-2.x** et de **pygtk** on commence par télécharger l'archive disponible ici ou ici, puis on la décompresse:

```
$ wget http://arpinux.org/public/pkgs/thelauncher-2.4.1.tar.gz
$ tar xvzf thelauncher-2.4.1.tar.gz
```

utilisation directe

vous pouvez constater qu'un executable “thelauncher” est déjà présent dans le dossier principal. vous pouvez utiliser ce script python en stand-alone en le copiant dans un dossier approprié

```
$ cd thelauncher-2.4.1/
$ cp thelauncher ~/bin/
```

et en l'assignant à un raccourcis clavier ou un lanceur de dock/panel.

utilisation dans le systray

hors gnome-panel, un applet de zone de notification est disponible dans le dossier ./thelauncher-2.4.1/tray-icon/ accompagné de son icone (vous pouvez modifier cette icone à votre convenance grâce à the Gimp ou Inkscape). pour l'insérer dans votre systray, il faut copier ce sript dans un dossier approprié et son icone dans /usr/share/pixmaps:

```
$ cd tray-icon/
$ cp thelauncher-trayicon ~/bin/
# cp thelauncher.svg /usr/share/pixmaps
```

il suffit ensuite de lancer thelauncher-trayicon dans votre systray au démarrage de votre session en le plaçant dans votre ~/.xinitrc:

```
## launch panel
pypanel &
## launch systray menu
thelauncher-trayicon &
```

vous pouvez aussi utiliser thelauncher-trayicon dans un systray indépendant.

utilisation commme applet gnome

si vous utilisez gnome-panel ou tout autre panel qui gère les applets gnome (comme xfce4-panel), l'installation est simplifiée: il suffit d'executer le script d'installation situé dans ./thelauncher-2.4.1/gnome-applet/ :

```
$ cd gnome-applet
# install-applet
```

et thelauncher sera disponible dans la liste de vos applets gnome.

conclusion

vous pouvez désormais avoir accès à un menu configurable (ou pas) sur n'importe quel gestionnaire de fenêtres. ou utiliser l'un de ces menus indépendants en complément de votre menu system par défaut, pour avoir vos applications favorites toujours accessibles par exemple.

---

contributeur: arpinux