

# DWM : the Dynamic Window Manager

**dwm** est écrit en C... pas de rafraichissement automatique .. tu veux configurer ? alors tu dois compiler :)

**dwm** a su rester simple .. vierge je dirais: vierge de tout gadget et autre options qui alourdissent l'environnement et finissent par nuire aux performances de l'ordinateur et de l'utilisateur, alors que ce n'était pas l'idée de départ. **dwm** est distribué par [suckless.org](http://suckless.org) *Dedicated to software that sucks less...* et adopte la philosophie KISS .

**dwm** est entièrement pilotable depuis le clavier mais sait aussi faire bon usage de votre souris (déplacement des clients, redimensionnement à la volée..).

si vous désirez tester dwm en live, allez faire un tour du côté du [livarp](#) ;) la version 0.3 a une session dwm pré-configurée et documentée...

## présentation

**DWM** est un tiling window manager, c'est à dire qu'il affiche vos fenêtres (clients) afin qu'elles occupent la totalité de la surface du bureau (tags). les tags occupés sont indiqués par un petit carré dans la barre de tags. les clients sont séparés en deux zones: le **Master**(client principal) et le **Stack** (clients empilés), organisés de différentes façons (layouts) pour optimiser la visibilité de vos applications ouvertes. la version de base dispose de trois layouts: Tile (Master à gauche et Stack à droite), Monocle (clients maximisés) ou NULL (no layout = clients libres), mais de nombreux patches existent pour ajouter des layouts.

## obtenir dwm

**Note:** *si vous désirez utiliser directement dwm sans changements de configuration, vous pouvez installer la version proposée par Debian: passer à la [section installation](#).*

pour récupérer dwm, plusieurs options s'offrent à vous.

- depuis le **site officiel** : <http://dwm.suckless.org>
- depuis les **sources Debian** : `$ apt-get source dwm`
- depuis le **dépôt git** : `$ git clone http://git.suckless.org/dwm`

## configuration

la configuration avant l'installation ? et oui!

comme dwm s'installe sans fichier de configuration, tous les changements doivent s'effectuer avant la compilation. ces modifications peuvent intervenir de deux façons:

- l'édition du fichier de configuration principal config.def.h et/ou
- l'application de patches sur dwm.c et config.h.

**Note:** si vous désirez utiliser directement dwm sans changements de configuration, vous pouvez installer la version proposée par votre distribution. passer à la [section installation](#).

## config.h

les modifications de base s'effectuent en copiant le fichier ./config.def.h en ./config.h et en l'éditant. je vous présente ici l'intégralité de ce fichier config.def.h commenté (notez que les raccourcis par défaut sont pour un clavier qwerty):

```
/* See LICENSE file for copyright and license details. */

/* appearance */
static const char font[] =
"-*-terminus-medium-r-*-*16-*-*-*-*-*"; /* sélectionner votre police
avec xfontsel*/
static const char normbordercolor[] = "#444444"; /* couleur du contour des
clients */
static const char normbgcolor[] = "#222222"; /* couleur de fond
clients-statusbar */
static const char normfgcolor[] = "#bbbbbb"; /* couleur du texte
clients-statusbar */
static const char selbordercolor[] = "#005577"; /* couleur de la bordure du
client au premier plan */
static const char selbgcolor[] = "#005577"; /* couleur du fond
sélectionné clients-statusbar */
static const char selfgcolor[] = "#eeeeee"; /* couleur du texte
clients-statusbar au premier plan */
static const unsigned int borderpx = 1; /* taille de la bordure des
clients en pixel(s) */
static const unsigned int snap = 32; /* distance d'adhérence en
pixel(s) pour les clients libres */
static const Bool showbar = True; /* afficher la statusbar */
static const Bool topbar = True; /* statusbar en haut de
l'écran */

/* tagging - nom et nombre de tags */
static const char *tags[] = { "1", "2", "3", "4", "5", "6", "7", "8", "9" };
```

```

/* Les règles (rules) spécifiques. utiliser xprop pour obtenir les infos. */
/* isfloating: True : client flottant, False : client en tiling */
/* tags mask: tag le client: 1 << 8 définit le tag 9, 1 << 7 définit le tag
8, ~0 définit tous les tags. */
/* sources: http://dwm.suckless.org/customisation/tagmask */
static const Rule rules[] = {
    /* xprop(1):
    *      WM_CLASS(STRING) = instance, class
    *      WM_NAME(STRING) = title
    */
    /* class      instance      title          tags mask      isfloating
monitor */
    { "Gimp",      NULL,          NULL,          0,             True,          -1
}, /* gimp s'ouvrira libre */
    { "Firefox",   NULL,          NULL,          1 << 8,        False,         -1
}, /* firefox s'ouvrira 'tilé' sur le tag 9 */
};

/* layout(s) */
static const float mfact      = 0.55; /* facteur de l'aire master
[0.05..0.95] */
static const int nmaster      = 1;     /* nombre de clients dans le master */
static const Bool resizehints = True;  /* respecte le redimensionnement
automatique des applications */

static const Layout layouts[] = {
    /* symbol      arrange function */
    { "[]=",       tile }, /* le première entrée sera le layout par
défaut de dwm: ici, tiling */
    { "><>",       NULL }, /* pas de layout = free layout */
    { "[M]",       monocle }, /* monocle = clients maximisés */
};

/* key definitions */
/* vous pouvez définir ici votre ou vos touches de modification. par défaut
défini à Mod1Mask (Alt) */
/* mais peut aussi être Mod4Mask (Super). vous pouvez également définir 2
modkey */
/* ex: #define MODKEY2 Mod4Mask définit la touche super comme seconde touche
de modification */
#define MODKEY Mod1Mask
#define TAGKEYS(KEY,TAG) \
    { MODKEY,             KEY,      view,           {.ui = 1
<< TAG} }, \
    { MODKEY|ControlMask, KEY,      toggleview,     {.ui = 1
<< TAG} }, \
    { MODKEY|ShiftMask,   KEY,      tag,           {.ui = 1
<< TAG} }, \
    { MODKEY|ControlMask|ShiftMask, KEY,      toggletag,     {.ui = 1
<< TAG} },

```

```

/* helper for spawning shell commands in the pre dwm-5.0 fashion */
#define SHCMD(cmd) { .v = (const char*[]){ "/bin/sh", "-c", cmd, NULL } }

/* commandes - raccourcis clavier */
static const char *dmenucmd[] = { "dmenu_run", "-fn", font, "-nb",
normbgcolor, "-nf", normfgcolor, "-sb", selbgcolor, "-sf", selfgcolor, NULL
};
static const char *termcmd[] = { "uxterm", NULL };

static Key keys[] = {
    /* modifier                                key          function      argument
*/
    { MODKEY,                                XK_p,         spawn,         {.v =
dmenucmd } },
    { MODKEY|ShiftMask,                      XK_Return,    spawn,         {.v =
termcmd } },
    { MODKEY,                                XK_b,         togglebar,     {0} },
    { MODKEY,                                XK_j,         focusstack,    {.i = +1
} },
    { MODKEY,                                XK_k,         focusstack,    {.i = -1
} },
    { MODKEY,                                XK_i,         incnmaster,    {.i = +1
} },
    { MODKEY,                                XK_d,         incnmaster,    {.i = -1
} },
    { MODKEY,                                XK_h,         setmfact,      {.f = -
0.05} },
    { MODKEY,                                XK_l,         setmfact,      {.f = +
0.05} },
    { MODKEY,                                XK_Return,    zoom,          {0} },
    { MODKEY,                                XK_Tab,       view,          {0} },
    { MODKEY|ShiftMask,                      XK_c,         killclient,    {0} },
    { MODKEY,                                XK_t,         setlayout,     {.v = &
layouts[0]} },
    { MODKEY,                                XK_f,         setlayout,     {.v = &
layouts[1]} },
    { MODKEY,                                XK_m,         setlayout,     {.v = &
layouts[2]} },
    { MODKEY,                                XK_space,     setlayout,     {0} },
    { MODKEY|ShiftMask,                      XK_space,     togglefloating, {0} },
    { MODKEY,                                XK_0,         view,          {.ui = ~0
} },
    { MODKEY|ShiftMask,                      XK_0,         tag,           {.ui = ~0
} },
    { MODKEY,                                XK_comma,     focusmon,      {.i = -1
} },
    { MODKEY,                                XK_period,    focusmon,      {.i = +1
} },
    { MODKEY|ShiftMask,                      XK_comma,     tagmon,        {.i = -1
} },

```

```

    { MODKEY|ShiftMask,          XK_period, tagmon,          {.i = +1
} },
    TAGKEYS(                      XK_1,          0)
    TAGKEYS(                      XK_2,          1)
    TAGKEYS(                      XK_3,          2)
    TAGKEYS(                      XK_4,          3)
    TAGKEYS(                      XK_5,          4)
    TAGKEYS(                      XK_6,          5)
    TAGKEYS(                      XK_7,          6)
    TAGKEYS(                      XK_8,          7)
    TAGKEYS(                      XK_9,          8)
    { MODKEY|ShiftMask,          XK_q,          quit,          {0} },
};

/* réactions à la souris */
/* click can be ClkLtSymbol, ClkStatusText, ClkWinTitle, ClkClientWin, or
ClkRootWin */
static Button buttons[] = {
    /* click          event mask      button      function
argument */
    { ClkLtSymbol,    0,          Button1,    setlayout,
{0} },
    { ClkLtSymbol,    0,          Button3,    setlayout,
{.v = &layouts[2]} },
    { ClkWinTitle,    0,          Button2,    zoom,
{0} },
    { ClkStatusText,  0,          Button2,    spawn,
{.v = termcmd } },
    { ClkClientWin,   MODKEY,      Button1,    movemouse,
{0} },
    { ClkClientWin,   MODKEY,      Button2,
togglefloating, {0} },
    { ClkClientWin,   MODKEY,      Button3,    resizemouse,
{0} },
    { ClkTagBar,      0,          Button1,    view,
{0} },
    { ClkTagBar,      0,          Button3,    toggleview,
{0} },
    { ClkTagBar,      MODKEY,      Button1,    tag,
{0} },
    { ClkTagBar,      MODKEY,      Button3,    toggletag,
{0} },
};

```

## francisation

les raccourcis clavier par défaut sont adaptés au clavier qwerty. pour adapter les raccourcis au clavier azerty, on modifie le fichier config.h:

remplacer:

```

TAGKEYS (          XK_1,          0)
TAGKEYS (          XK_2,          1)
TAGKEYS (          XK_3,          2)
TAGKEYS (          XK_4,          3)
TAGKEYS (          XK_5,          4)
TAGKEYS (          XK_6,          5)
TAGKEYS (          XK_7,          6)
TAGKEYS (          XK_8,          7)
TAGKEYS (          XK_9,          8)

```

par

```

TAGKEYS (          XK_ampersand,    0)
TAGKEYS (          XK_eacute,       1)
TAGKEYS (          XK_quotedbl,     2)
TAGKEYS (          XK_apostrophe,   3)
TAGKEYS (          XK_parenleft,    4)
TAGKEYS (          XK_minus,        5)
TAGKEYS (          XK_egrave,       6)
TAGKEYS (          XK_underscore,   7)
TAGKEYS (          XK_ccedilla,     8)

```

et remplacer

```

{ MODKEY,          XK_0,          view,          {.ui = ~0 } },
{ MODKEY|ShiftMask, XK_0,          tag,          {.ui = ~0 } },

```

par

```

{ MODKEY,          XK_agrave,      view,          {.ui = ~0 }
},
{ MODKEY|ShiftMask, XK_agrave,      tag,          {.ui = ~0 }
},

```

## patches

les patches sont des modifications qui permettent d'ajouter des fonctionnalités à dwm. sur [suckless.org](http://suckless.org), les patches sont réunis sur [cette page](#). libre à vous de les adapter à vos besoins. pour appliquer un patch:

- utilisateur de git:cd dwm-directory  
git diff > dwm-X.Y-yourpatchname.diff
- utilisateur de l'archive:cd dwm-directory  
patch -p1 < path/to/patch.diff

# installation

## paquet officiel

pour les utilisateurs Debian et dérivées, si vous souhaitez utiliser la version de base de DWM sans configuration particulière, vous pouvez simplement utiliser la commande suivante qui installera dwm et quelques outils de chez suckless.org (dmenu, tabbed..):

```
# apt-get install dwm suckless-tools
```

dwm s'installera dans /usr/bin.

## depuis les sources git

si vous avez téléchargé les sources depuis les dépôts git, vous devez installer les dépendances:

```
# apt-get install libx11-dev libxinerama-dev
```

la procédure d'installation est classique: une fois les modifications et les patches appliqués, il vous suffit de vous placer dans le répertoire de dwm puis de lancer:

```
$ make  
# make install
```

dwm s'installera par défaut dans /usr/local/bin.

## depuis les sources Debian

**sources:** l'excellent [post de thuban](#).

On récupère les sources ainsi :

```
$ apt-get source dwm
```

On se déplace dans le dossier des sources. Avec le caractère "\*", on n'a pas besoin de préciser la version de dwm :

```
cd dwm*
```

Attention: Si vous voulez configurer dwm, éditez ici le config.def.h, pas le config.h (c'est la petite exception debian).

Une fois satisfaits, reconstruisez le paquet :

```
# dpkg-buildpackage -rfakeroot -uc -b
```

Un paquet est alors disponible dans le répertoire parent, sous la forme dwm- version.deb. Pour vous assurer d'avoir les dépendances nécessaires, il suffit de lancer :

```
# apt-get build-dep dwm
```

Pour installer le paquet créé :

```
# dpkg -i dwm*.deb
```

**Note**: Afin d'éviter toute mise à jour non souhaitée de dwm, écrivez ceci dans le fichier /etc/apt/preferences :

```
Package: dwm
Pin: release a=now
Pin-Priority: 1001
```

Où alors lancer :

```
# aptitude hold dwm
```

## utilisation

### lancement

dwm se lance comme une session classique: soit par gdm ou tout autre gestionnaire de session graphique, soit par startx. voici mon ~/.xinitrc pour l'exemple d'un lancement en startx:

#### xinitrc

```
#!/bin/bash
#####
# ~/.xinitrc by arpinux 2011 #
#####
## D-Bus ##
if which dbus-launch >/dev/null && test -z "$DBUS_SESSION_BUS_ADDRESS";
then
    eval "$(dbus-launch --sh-syntax --exit-with-session)"
fi
## trackpad ## tapbutton off by default ##
synclient VertTwoFingerScroll=1
```



```

synclient HorizTwoFingerScroll=1
synclient TapButton1=0
## dualscreen ## edit if needed ##
xrandr --output LVDS --mode 1024x768 --pos 0x0 --rotate normal --output
VGA-0 --mode 1024x768 --pos 1024x0 --rotate normal
## read ~/.Xresources file
xrdb -merge ~/.Xresources
## set wallpaper / color the screen ##
#xsetroot -solid grey20 &
feh --no-xinerama --bg-scale ~/.arp_setups/bg.png ## uncomment to
display default wallpaper ##
#nitrogen --restore & ## uncomment to run your personal wallpaper ##
## launch terminal daemon ##
urxvtd -q- -f -o
## set session-killer
setxkbmap -option terminate:ctrl_alt_bksp
## launch session
exec ck-launch-session /usr/local/bin/dwm

```

## raccourcis clavier

souvenez-vous que **dwm s'organise autour de tags**, l'équivalent des *bureaux virtuels*, mais avec plus de possibilités:

on peut rendre un client visible sur tous les tags avec [Alt]+[Shift]+[0] (l'équivalent du "sticky" pour les bureaux virtuels), mais on peut aussi assigner 2 tags à un client, ce qui reviendrait à afficher une fenêtre sur 2 bureaux virtuels seulement.

la fonction "exposé" est présente dans dwm ... oO en effet, le raccourcis [Alt]+[0] affiche tous les clients sur le tag courant, il suffit d'afficher un autre tag pour rétablir l'affichage normal.

voici quelques commandes de bases pour commencer à utiliser dwm. 🚨 si vous avez modifié les raccourcis clavier dans config.h, il faudra adapter. la touche de modification *Mod1* par défaut est [Alt].

## les raccourcis clavier

modifier	touche	description
Mod1+Shift	return	ouvre le terminal par défaut
Mod1	,	va à l'écran précédent
	.	va à l'écran suivant
Mod1+Shift	,	deplace le client vers l'écran précédent
	.	deplace le client vers l'écran suivant

Mod1	b	affiche/masque la barre d'info
	t	definir le layout en "tile" (mosaïque)
	f	definir le layout en "floating" (libre)
	m	definir le layout en "monocle" (maximisé)
	space	échanger entre le layout actuel et le précédent
	j	donne le focus au client suivant
	k	donne le focus au client précédent
	h	agrandi la taille du master
	l	diminue la taille du master
	return	échange le client ayant le focus avec celui dans le master
Mod1+Shift	c	fermer le client ayant le focus
	space	libérer/"tiler" le client ayant le focus
	0	mettre le client qui a le focus sur tous les tags
Mod1	1,2..n	afficher les clients du tag n
	0	afficher les clients de tous les tags (fonction "exposé")
Mod1+Shift	q	quitter dwm

### le comportement de la souris sur les clients

*rappel: b1=clic gauche, b2=clic central, b3=clic droit*

modifier	bouton	description
Mod1	bouton1	déplacer la fenêtre au glissé, fait passer la fenêtre en mode "libre"
	bouton2	libérer/"tiler" le client
	bouton3	redimensionne le client depuis le coin inférieur droit, fait passer la fenêtre en mode "libre"

### le comportement de la souris sur les tags

modifier	bouton	description
	bouton1	affiche les clients du tag sélectionné (va au tag sélectionné)
	bouton3	affiche/masque les clients du tag sélectionné sur le tag courant
Mod1	bouton1	applique le tag sélectionné au client qui a le focus (deplace le client vers le tag sélectionné)
	bouton3	ajoute/enlève le tag sélectionné au client qui a le focus

### le comportement sur le bouton de layout

bouton	description
bouton1	permuter les layouts entre "tile"(mosaïque) et "floating"(libre)
bouton3	sélectionner le tag "floating" (libre)

## personnalisation

### conky dans la statusbar

la statusbar de dwm indique les tags, le layout et la version de dwm utilisée. mais on peut lui faire afficher beaucoup plus. pour cela, plusieurs possibilités: conky, dzen2, un script bash, un exécutable en C...

pour tester, lancer dans une console:

```
xsetroot -name "`date`"
```

Dans le cas de conky, il ne s'affiche pas réellement: il s'exécute et est "lu" par la statusbar. Pour cela, vous aurez besoin d'un conkyrc particulier car il indiquera à conky de ne rien afficher !! ... En fait, les informations seront envoyées vers STDOUT et récupérées par dwm. Voici comment procéder:

```
#####
# 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
!${cpu}%- ${acpitemp}°\
!${memperc}%- $mem\
!${fs_used_perc /}%- ${diskio}/s- ${ibm_temps 2}°\
!${if_match "$ibm_volume" == "mute" }M${else}${ibm_volume}/14${endif}\
!${time %d/%m %I:%M}!
```

Vous constatez que les fonctions sont réduites lorsque conky ne s'affiche pas dans X. De plus, les informations 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 &
```

différents patches permettent de colorer la statusbar, de la rendre cliquable etc...

## dwm\_reloaded

j'utilise dwm au quotidien et dans [livarp](#). la version est patchée et propose toute une série de modifications et raccourcis clavier.

pour obtenir **dwm\_reloaded**, utiliser les dépôts [gitorious](#):

```
$ git clone git://gitorious.org/livarp/dwm_reloaded.git
```

ce dépôt propose deux versions fr/gb et s'installe de [façon classique](#).

pour une présentation/documentation complète sur dwm\_reloaded, visiter la page [d'aide du livarp dédiée](#).

## conclusion

DWM est mon gestionnaire de fenêtres préféré! c'est aussi le gestionnaire de fenêtres par défaut du livarp. depuis que j'ai commencé à l'utiliser/le modifier, il n'a planté qu'une fois. dwm est léger, efficace, gère les écrans multiples, est simple à configurer. dwm conviendra parfaitement aux pro soucieux d'optimiser leur espace de travail comme aux g33ks voulant mettre les doigts dans la console ;) .

From:

<http://arpinux.org/x/> - **arpinux wiki**

Permanent link:

<http://arpinux.org/x/doku.php/wms:dwmm>

Last update: **2013/04/25 12:32**

