Initram: LAB2 - busybox avec environnement graphique

Objet	Création d'un INITRAM avec busybox et X11
Niveau requis	débutant, avisé
Débutant, à savoir	
Suivi	:DONE:

X est un programme installé sur une machine Linux avec un moniteur (les serveurs n'utilisent donc généralement pas X). Le travail de X consiste à communiquer avec le noyau Linux au nom des programmes à interface graphique.

Au lieu d'exécuter X, on peut exécuter une version différente capable de créer des affichages virtuels (par exemple, pour exécuter une application graphique sans interface dans un conteneur Docker): Xvfb (framebuffer virtuel).

Ce Lab décrit l'installation et la configuration du système ramebuffer virtuel sur l'image de base Busybox. Nous allons installer, les bibliothèques pour les polices, expat, XML, un server graphique (Xvfb), divers petits outils, un gestionnaire de fenêtres (JWM), les bibliothèques JPEG et le navigateur web Links (en français).

Préparation du rootfs

Charger la structure de répertoire de rootfs

Charger la structure du répertoire rootfs avec l'image de base créée dans le LAB1.

```
export CLFS="/var/lfs/x11-busybox"
mkdir -pv ${CLFS}/{rootfs,_src,_pkg}
cd ${CLFS}/rootfs
zcat /var/lfs/x86-busybox/rootfs.gz | cpio -idmv --no-absolute-filenames
```

A propos de ldconfig

Lorsqu'un logiciel requiert une liaison symbolique des bibliothèques il faut mettre à jour le cache.

La commande `ldconfig` est utilisée pour créer les liens nécessaires et mettre en cache les bibliothèques partagées les plus récentes trouvées dans les répertoires indiqués sur la ligne de commande, dans le fichier `/etc/ld.so.conf`, et dans les répertoires sûrs (`/lib` et `/usr/lib`). Le cache est utilisé par le chargeur/éditeur de liens `ld.so` ou `ld-linux.so`.

```
cp /etc/ld.so.conf* ${CLFS}/rootfs/etc -R
```

```
cp /sbin/ldconfig ${CLFS}/rootfs/sbin
```

Installation et config du système X

Serveur Xvfb d'XFree86, configuration des polices et des locales.

expat-2.0.0 - XML parser library

Expat (http://expat.sourceforge.net/) contient des bibliothèques d'analyse XML:

```
cd ${CLFS}/_src
wget
https://sourceforge.net/projects/expat/files/expat/2.0.0/expat-2.0.0.tar.gz/
download -O expat-2.0.0.tar.gz
tar xzf expat-2.0.0.tar.gz
cd expat-2.0.0
./configure --sysconfdir=/etc --prefix=/usr \
   --mandir=/usr/share/man
make
make DESTDIR=$PWD/_pkg install
strip -v _pkg/usr/lib/*
strip -v _pkg/usr/bin/*
cd _pkg/usr/lib
ln -s libexpat.so.1.5.0 libexpat.so.0
cp -a *.so* ${CLFS}/rootfs/usr/lib
cd ..
cp -a bin/* ${CLFS}/rootfs/usr/bin
cd ${CLFS}
```

freetype-2.3.1 - Bibliothèques de police sytème

Le paquet freetype (http://www.freetype.org/) contient des bibliothèques utilisées par X pour la configuration des polices du système:

```
cd ${CLFS}/_src
wget
https://sourceforge.net/projects/freetype/files/freetype2/2.3.1/freetype-2.3
.1.tar.gz/download -O freetype-2.3.1.tar.gz
tar xzf freetype-2.3.1.tar.gz
cd freetype-2.3.1
./configure --sysconfdir=/etc --prefix=/usr \
   --mandir=/usr/share/man
make
```

```
make DESTDIR=$PWD/_pkg install
strip -vs _pkg/usr/lib/*
cp -a _pkg/usr/bin/* ${CLFS}/rootfs/usr/bin
cp -a _pkg/usr/lib/*.so* ${CLFS}/rootfs/usr/lib
```

Faire un 'ldd' sur xmlwf pour récupérer la liste des bibliothèques:

```
ldd ${CLFS}/rootfs/usr/bin/
```

```
    libexpat.so.1 => /lib/libexpat.so.1 (0x00007f4dd4b8c000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f4dd47bf000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f4dd4dca000)
```

Copier si nécessaire les librairies dans le rootfs

```
cp -avLfn /lib/libexpat.so.1 ${CLFS}/rootfs/lib
cp -avLfn /lib64/{libc.so.6,ld-linux-x86-64.so.2} ${CLFS}/rootfs/lib64
```

fontconfig 2.4.2 - Outils de gestion des polices

Le paquet fontconfig (www.fontconfig.org/wiki/) fourni la bibliothèque libfontconfig, utilisée par de nombreux programme sous X.

Note: XFree86 fournit aussi ces bibliothèques, et les utilitaires. On a choisi d'utiliser le paquet original, car il fonctionne mieux avec JWM:

```
cd ${CLFS}/_src
wget http://fontconfig.org/release/fontconfig-2.4.2.tar.gz
tar xzf fontconfig-2.4.2.tar.gz
cd fontconfig-2.4.2
./configure --sysconfdir=/etc --prefix=/usr \
   --mandir=/usr/share/man --localstatedir=/var
make
make DESTDIR=$PWD/_pkg install
strip -v _pkg/usr/bin/*
strip -v _pkg/usr/lib/*
cp -a _pkg/usr/bin/* ${CLFS}/rootfs/usr/bin
cp -a _pkg/usr/lib/*.so* ${CLFS}/rootfs/usr/lib
cp -a _pkg/etc ${CLFS}/rootfs
cp -a _pkg/var ${CLFS}/rootfs
cd ${CLFS}
```

Faire un 'ldd' sur fc-cache pour récupérer la liste des bibliothèques:

```
```

```
ldd ${CLFS}/rootfs/usr/bin/fc-cache
```

```
    libfreetype.so.6 => /lib64/libfreetype.so.6 (0x00007f96a83c5000)
    libexpat.so.1 => /lib/libexpat.so.1 (0x00007f96a819b000)
    libfontconfig.so.1 => /lib64/libfontconfig.so.1 (0x00007f96a7f58000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f96a7b8b000)
    libz.so.1 => /lib64/libz.so.1 (0x00007f96a7975000)
    libbz2.so.1 => /lib64/libbz2.so.1 (0x00007f96a7764000)
    libpng15.so.15 => /lib64/libpng15.so.15 (0x00007f96a7539000)
    libuuid.so.1 => /lib64/libuuid.so.1 (0x00007f96a7334000)
    libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f96a7117000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f96a8698000)
    libm.so.6 => /lib64/libm.so.6 (0x00007f96a6e15000)
```

Copier si nécessaire les librairies dans le rootfs

```
cp -avLn /lib/libexpat.so.1 ${CLFS}/rootfs/lib
cp -avLn
/lib64/{libfreetype.so.6,libfontconfig.so.1,libc.so.6,libz.so.1,libbz2.so.1,
libpng15.so.15,libuuid.so.1,libpthread.so.0,ld-linux-x86-64.so.2,libm.so.6}
${CLFS}/rootfs/lib64
```

Xfree86 - Serveur X

Nous allons utiliser la versions binaires du serveur Xvfb et les polices distribuée par Xfree86.org (www.free86.org/).

Xvfb - Serveur X factice

Le serveur X « factice » de mémoire d'image (framebuffer) virtuelle fournit un serveur X pouvant fonctionner sur des machines sans écran et sans périphérique physique d'entrée.

```
cd ${CLFS}/_src
mkdir -p XFree86-4.6.0 && cd XFree86-4.6.0
wget
ftp://ftp.xfree86.org/pub/XFree86/4.6.0/binaries/Linux-amd64-glibc23/Xvfb.tg
z
tar xzf Xvfb.tgz
cp bin/Xvfb ${CLFS}/rootfs/usr/bin
strip ${CLFS}/rootfs/usr/bin/Xvfb
chmod 4711 ${CLFS}/rootfs/usr/bin/Xvfb
cd ${CLFS}
```

Faire un 'ldd' sur fc-cache pour récupérer la liste des bibliothèques:

```
ldd ${CLFS}/rootfs/usr/bin/Xvfb
```

```
    libfreetype.so.6 => /lib64/libfreetype.so.6 (0x00007f11335e6000)
    libz.so.1 => /lib64/libz.so.1 (0x00007f11333d0000)
    libm.so.6 => /lib64/libm.so.6 (0x00007f11330cd000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f1132d00000)
    libbz2.so.1 => /lib64/libbz2.so.1 (0x00007f1132af0000)
    libpng15.so.15 => /lib64/libpng15.so.15 (0x00007f11328c4000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f11338b9000)
```

Copier les librairies dans le rootfs

```
cp -avLn
/lib64/{libfreetype.so.6,libz.so.1,libm.so.6,libc.so.6,libbz2.so.1,libpng15.
so.15,ld-linux-x86-64.so.2} ${CLFS}/rootfs/lib64
```

rgb.txt - Les couleurs RGB sous X

Le fichier de configuration des couleurs utilisée par le serveur X se nomme rgb.txt, nous vous proposons de copier celui du système hôte. La bibliothèque `libX11.so` d'Xorg va chercher les fichiers de configuration dans `/usr/share/X11`, et le serveur Xvesa dans `/usr/X11R6/lib/X11`, créer un lien dans `/usr/share/X11` afin de satisfaire cela:

```
mkdir -p ${CLFS}/rootfs/usr/share/X11
cp /usr/X11R6/lib/X11/rgb.txt ${CLFS}/rootfs/usr/share/X11
```

Passer en chroot dans le `rootfs`:

```
/sbin/chroot ${CLFS}/rootfs /bin/ash
```

Dans le chroot créer un lien vers rgb.txt

```
mkdir -p /usr/X11R6/lib/X11/
ln -s /usr/share/X11/rgb.txt /usr/X11R6/lib/X11/rgb.txt
exit
```

Xfnts - Les polices

Pour que le serveur fonctionnent il faut les polices de base, que l'on peut télécharger depuis xfree86.org, les compiler depuis les paquets d'Xorg, ou les copier depuis le système hôte. Les polices du système peuvent se mettre dans différents dossiers, et une fois installées il faut lancer `lc-cache` pour mettre à jour le cache. Attention les fonts prennent de la place, on peut ne copier que le minimum.

```
cd ${CLFS}/_src/XFree86-4.6.0
wget
http://ftp.xfree86.org/pub/XFree86/4.6.0/binaries/Linux-ix86-glibc23/Xfnts.t
gz
wget
ftp://ftp.xfree86.org/pub/XFree86/4.6.0/binaries/Linux-amd64-glibc23/Xf100.t
gz
tar xzf Xfnts.tgz
tar xzf Xf100.tgz
mkdir -p ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts
mkdir -p ${CLFS}/rootfs/usr/share/fonts/
cp -a lib/X11/fonts/* ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts
cp -a /usr/share/fonts/* ${CLFS}/rootfs/usr/share/fonts/
cd ${CLFS}
```

Pour regénérer le fichier fonts.dir, il vous faut lancer `mkfontdir` sur le réperoire en question:

```
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/75dpi
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/100dpi
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/CID
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/local
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/misc
mkfontdir ${CLFS}/rootfs/usr/X11R6/lib/X11/fonts/util
```

Les fichiers de configuration de fontconfig, se trouvent dans `/etc/fonts`, fourni par le paquet fontconfig.

Passer en chroot dans le rootfs:

```
/sbin/chroot ${CLFS}/rootfs /bin/ash
```

Dans le chroot lancer 'fc-cache', et pour connaître la liste des polices utiliser 'fc-list'.

```
/sbin/ldconfig
fc-cache -v
fc-list
exit
```

Xlib locale - Les fichiers de localisation

On peut copier les fichiersde localisation depuis le système hôte ou utiliser les fichiers distribuées par XFree86. Exemple de copie de toutes les locales iso8859-1, iso8859-2, et iso8859-15 depuis le système hôte:

- ISO 8859-1 (Latin-1 ou Europe occidentale) prend en charge le support partiel du français
    - ISO 8859-2 (latin-2 ou européen central) prend en charge les langues d'Europe centrale ou de l'Est basées sur un alphabet romain

- ISO 8859-15 (Latin-9) prend en charge le support du français,

```
mkdir -p ${CLFS}/rootfs/usr/share/X11/locale cp -a
/usr/share/X11/locale/{iso8859-1,iso8859-15,iso8859-2}
${CLFS}/rootfs/usr/share/X11/locale
```

Utilisation de X

A noter que l'on peut déjà utiliser Xvfb comme terminal X, si vous avez une machine sur le réseau acceptant les connexions Xdmcp. Pour cela on peut lancer le pseudo serveur ainsi:

```
Xvfb :99 -screen 0 640x480x8
```

libpng-1.2.18 - Bibliothèques PNG

Les bibliothèques PNG (http://libpng.org/pub/png/libpng.html)permettent de manipuler et d'utiliser les images au format .png:

```
cd ${CLFS}/_src
wget
https://sourceforge.net/projects/libpng/files/libpng12/older-releases/1.2.18
/libpng-1.2.18.tar.bz2/download -O libpng-1.2.18.tar.bz2
tar xjf libpng-1.2.18.tar.bz2
cd libpng-1.2.18
./configure --enable-shared --prefix=/usr \
   --mandir=/usr/share/man
make
make DESTDIR=$PWD/_pkg install
strip _pkg/usr/lib/*.so*
cp -a _pkg/usr/lib/libpng12.so* ${CLFS}/rootfs/usr/lib
cp -a _pkg/usr/bin/libpng12* ${CLFS}/rootfs/usr/bin
cd ${CLFS}
```

jpeg-6b - Bibliothèques JPEG

Les bibliothèques de manipulation des images JPEG, et quelques petits utilitaires:

```
cd ${CLFS}/_src
wget http://www.ijg.org/files/jpegsrc.v6b.tar.gz
tar xzf jpegsrc.v6b.tar.gz
cd jpeg-6b
./configure --enable-shared --prefix=/usr \
   --mandir=/usr/share/man
```

```
sed -i -e "s/\.\/libtool/libtool/g" Makefile
make
strip .libs/*
cp -a .libs/*.so* ${CLFS}/rootfs/usr/lib
cp .libs/{cjpeg,djpeg,jpegtran} ${CLFS}/rootfs/usr/bin
cd ${CLFS}
```

tiff-3.8.2 - Bibliothèques et utilitaires TIFF

Les bibliothèques de manipulation des images TIFF, et quelques petits utilitaires en option:

```
cd ${CLFS}/_src
wget
https://src.fedoraproject.org/repo/pkgs/libtiff/tiff-3.8.2.tar.gz/fbb6f446ea
4ed18955e2714934e5b698/tiff-3.8.2.tar.gz
tar xzf tiff-3.8.2.tar.gz
cd tiff-3.8.2
./configure  --prefix=/usr --mandir=/usr/share/man
make
make DESTDIR=$PWD/_pkg install
strip _pkg/usr/bin/*
strip _pkg/usr/lib/*.so*
cp -a _pkg/usr/lib/*.so* ${CLFS}/rootfs/usr/lib
cd ${CLFS}
```

links-2.20.1 - Navigateur web graphique et texte

Links (links.twibright.com) est un navigateur web proposant un mode texte et un mode graphique, il est traduit dans de multiples langues, dont le français:

```
cd ${CLFS}/_src
wget http://links.twibright.com/download/links-2.20.1.tar.bz2
tar xjf links-2.20.1.tar.bz2
cd links-2.20.1
./configure --prefix=/usr --sysconfdir=/etc --mandir=/usr/share/man \
   --without-directfb --without-ssl --enable-graphics
make
make DESTDIR=$PWD/_pkg install
strip -v _pkg/usr/bin/*
cp -v _pkg/usr/bin/* ${CLFS}/rootfs/usr/bin
cd ${CLFS}
```

Faire un 'ldd' sur jwm pour récupérer la liste des bibliothèques:

```

```

```
ldd ${CLFS}/rootfs/usr/bin/Xvfb
```

```
    libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f827ee09000)
    libpng15.so.15 => /lib64/libpng15.so.15 (0x00007f827ebde000)
    libfontconfig.so.1 => /lib64/libfontconfig.so.1 (0x00007f827e99b000)
    libfreetype.so.6 => /lib64/libfreetype.so.6 (0x00007f827e6dc000)
    libX11.so.6 => /lib64/libX11.so.6 (0x00007f827e39e000)
    libbz2.so.1 => /lib64/libbz2.so.1 (0x00007f827e18d000)
    libz.so.1 => /lib64/libz.so.1 (0x00007f827df77000)
    libm.so.6 => /lib64/libm.so.6 (0x00007f827dc75000)
    libgomp.so.1 => /lib64/libgomp.so.1 (0x00007f827da4e000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f827d681000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f827f039000)
    libexpat.so.1 => /lib/libexpat.so.1 (0x00007f827d457000)
    libuuid.so.1 => /lib64/libuuid.so.1 (0x00007f827d251000)
    libxcb.so.1 => /lib64/libxcb.so.1 (0x00007f827d029000)
    libdl.so.2 => /lib64/libdl.so.2 (0x00007f827ce25000)
    libXau.so.6 => /lib64/libXau.so.6 (0x00007f827cc20000)
```

Copier les librairies dans le rootfs

```
cp -avLn
/lib64/{libpthread.so.0,libpng15.so.15,libfontconfig.so.1,libfreetype.so.6,l
ibX11.so.6,libbz2.so.1,libz.so.1,libm.so.6,libgomp.so.1,libc.so.6,ld-linux-
x86-64.so.2,libuuid.so.1,libxcb.so.1,libdl.so.2,libXau.so.6}
${CLFS}/rootfs/lib64
cp -avLn /lib/libexpat.so.1 ${CLFS}/rootfs/lib
```

jwm-2.0 - Gestionnaire de fenêtres

Joe's Window Manager (http://www.joewing.net/programs/jwm/) est un gestionnaire de fenêtres ultra léger, et convivial. Le fichier de configuration principal est dans `/etc/jwm/system.jwmrc`, comprenant la config du menu et du style:

```
cd ${CLFS}/_src
wget
https://sourceforge.net/projects/jwm/files/jwm/2.1.0/jwm-2.1.0.tar.bz2/downl
oad -O jwm-2.1.0.tar.bz2
tar xjf jwm-2.1.0.tar.bz2
cd jwm-2.1.0
./configure --prefix=/usr --mandir=/usr/share/man \
   --sysconfdir=/etc/jwm --disable-xinerama
make
strip src/jwm
cp src/jwm ${CLFS}/rootfs/usr/bin
mkdir ${CLFS}/rootfs/etc/jwm
```

```
cp example.jwmrc ${CLFS}/rootfs/etc/jwm/system.jwmrc
cd ${CLFS}
```

Faire un 'ldd' sur jwm pour récupérer la liste des bibliothèques:

```
ldd ${CLFS}/rootfs/usr/bin/jwm
```

```
    libX11.so.6 => /lib64/libX11.so.6 (0x00007fdbf30cf000)
    libpng15.so.15 => /lib64/libpng15.so.15 (0x00007fdbf2ea4000)
    libXft.so.2 => /lib64/libXft.so.2 (0x00007fdbf2c8d000)
    libXrender.so.1 => /lib64/libXrender.so.1 (0x00007fdbf2a82000)
    libXext.so.6 => /lib64/libXext.so.6 (0x00007fdbf2870000)
    libc.so.6 => /lib64/libc.so.6 (0x00007fdbf24a2000)
    libxcb.so.1 => /lib64/libxcb.so.1 (0x00007fdbf227a000)
    libdl.so.2 => /lib64/libdl.so.2 (0x00007fdbf2076000)
    libz.so.1 => /lib64/libz.so.1 (0x00007fdbf1e5f000)
    libm.so.6 => /lib64/libm.so.6 (0x00007fdbf1b5d000)
    libfontconfig.so.1 => /lib64/libfontconfig.so.1 (0x00007fdbf191b000)
    libfreetype.so.6 => /lib64/libfreetype.so.6 (0x00007fdbf165b000)
    /lib64/ld-linux-x86-64.so.2 (0x00007fdbf3421000)
    libXau.so.6 => /lib64/libXau.so.6 (0x00007fdbf1457000)
    libuuid.so.1 => /lib64/libuuid.so.1 (0x00007fdbf1027000)
    libpthread.so.0 => /lib64/libpthread.so.0 (0x00007fdbf0e0b000)
    libbz2.so.1 => /lib64/libbz2.so.1 (0x00007fdbf0bfb000)
```

Copier les librairies dans le rootfs

```
cp -avLn
/lib64/{libX11.so.6,libpng15.so.15,libXft.so.2,libXrender.so.1,libXext.so.6,
libc.so.6,libxcb.so.1,libdl.so.2,libz.so.1,libm.so.6,libfontconfig.so.1,libf
reetype.so.6,ld-linux-
x86-64.so.2,libXau.so.6,libuuid.so.1,libpthread.so.0,libbz2.so.1}
${CLFS}/rootfs/lib64
```

On peut démarrer le pseudo server X et JWM avec la commande ci-dessous, ou en créant un script, tel que `/usr/bin/startx` avec pour contenu:

```
Xvfb :99 -screen 0 640x480x8 & exec jwm
```

Génération des images

Génération de l'initramfs

Créer un nouvelle image initramfs

```
cd ${CLFS}/rootfs
find . -print | cpio -o -H newc | gzip -9 > ../rootfs.gz
cd ${CLFS}
```

Génération d'un ISO

Pour créer une nouvelle image ISO copier le dossier `rootcd` créé dans le LAB1, copier initramfs dans le /boot de la racine de cd-rom (rootcd), et pour finir créer un image ISO avec `genisoimage`:

```
cp rootfs.gz rootcd/boot
genisoimage -R -o minix86-cooking.iso -b boot/isolinux/isolinux.bin \
   -c boot/isolinux/boot.cat -no-emul-boot -boot-load-size 4 \
   -V "MiniX86" -boot-info-table rootcd
```

Utilisation de Xvfb

Création d'un moniteur virtuel

Au lieu d'exécuter X, on utiliser var créer des affichages virtuels: Xvfb (tampon de mémoire virtuelle):

```
Xvfb :1 -screen 0 1024x768x16
```

Cela démarrera le serveur Xvfb avec un affichage **1** et un écran virtuel (moniteur) **0**. On peut y accéder en tapant simplement `DISPLAY=:1.0` avant d'exécuter un programme graphique, auquel cas le programme démarrera dans l'écran virtuel.

Pour s'assurer que l'écran est toujours en cours d'exécution taper:

```
ps aux | grep X
```

```
root      1436  3.1  0.7 684580 91144 tty7     Ssl+ 09:47   3:31 /usr/bin/X
-core :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -
novtswitch
manos    22018  0.0  0.1 164960 20756 pts/27   Sl+  11:37   0:00 Xvfb :1 -
screen 0 1024x768x16
```

On voit que l'affichage normal **0** (Une façon de dire que c'est l'écran par défaut) fonctionne entoujours. On peut également voir le deuxième affichage: **1** et l'écran **0** avec une résolution de 1024 × 768.

Pour l'utiliser, ouvrir un nouveau terminal et taper:

```
DISPLAY=:1.0 firefox
```

On utilise DISPLAY sur la même ligne pour s'assurer que le sous-processus hérite de la variable DISPLAY. Mais pour cela, on peut également taper:

```
DISPLAY=:1.0
export DISPLAY
firefox
```

Exécuter un programme graphique dans un conteneur Docker

On peut créer un écran virtuel dans un conteneur Docker:

```
docker run -it ubuntu bash
```

```
root@660ddd5cc806:/# apt-get update
root@660ddd5cc806:/# apt-get install xvfb
root@660ddd5cc806:/# Xvfb :1 -screen 0 1024x768x16 &> xvfb.log  &
root@660ddd5cc806:/# ps aux | grep X
root        11  0.0  0.1 169356 20676 ?        Sl   10:49   0:00 Xvfb :1 -
screen 0 1024x768x16
```

Maintenant que nous l'écran virtuel fonctionne, on peut exécuter quelque chose de graphique dessus.

Tout d'abord, définir la variable `display` et utiliser `export` pour s'assurer que tous les sous-shells ou sous-processus utilisent le même affichage (avec export, ils héritent de la variable DISPLAY!):

```
DISPLAY=:1.0
export DISPLAY
```

Maintenant, on peut simplement exécuter le navigateur

```
firefox
```

```
 GLib-CRITICAL **: g_slice_set_config: assertion 'sys_page_size == 0' failed
Xlib:  extension "RANDR" missing on display ":99.0".
 GConf-WARNING **: Client failed to connect to the D-BUS daemon:
//bin/dbus-launch terminated abnormally without any error message
```