Rapport de l’installation de Gimp de Clément MONFORT

Préambule

Bien que je sois très familier avec l'usage de gimp, comme la plupart des utilisateurs j'utilise un gestionnaire de paquets pour gérer les dépendances, cependant quand on souhaite contribuer aux projets il est préférable de compiler les sources pour pouvoir tester ses modifications.
il est bon de noter que j'ai réalisé l'installation sur un **Ubuntu DESKTOP 22.04.3 LTS**, gimp est pensé pour tout d'abord pour linux qui l'environement de développement principale, d'où mon choix de **Ubuntu** la distribution la plus répandue sous **Linux**.

Installation des dépendances

Sous les distribution **Debian** comme **Ubuntu**, il est possible d'installer en une seule commande avec le gestionnaire de paquets **apt** :

```
sudo sh -c 'apt build-dep gimp && apt install meson'
```

Si comme moi cette erreure s'affiche `E: You must put some 'deb-src' URIs in your sources.list`, il vous faudra faire les commandes suivantes avant de réessayer :

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list~
sudo sed -Ei 's/^# deb-src /deb-src /' /etc/apt/sources.list
sudo apt update
```

Une commande similaire existe pour d'autre OS comme **Fedora** :

```
sudo dnf builddep gimp
```

Une autre solution est d'installer toute les dépendances de la version de Gimp que vous voulez utilisez à la main, voici un l'exemple présenter dans la documentation officiel (non à jour et seulement pour **Ubuntu**) :

```
sudo sh -c ‘apt update && apt install libtool intltool meson
asciidoc exiv2 libgexiv2-dev gjs gtk-3-examples gtk-doc-tools
jasper libaa1-dev libappstream-glib-dev libarchive-dev
libavcodec-dev libavformat-dev libavutil-dev libbz2-dev libgs-dev
libgtk-3-dev libgtk2.0-dev gobject-introspection
libgirepository1.0-dev libgudev-1.0-dev libheif-dev libjson-c-dev
libjson-glib-dev liblcms2-dev liblensfun-dev libmng-dev
libopenexr-dev libjpeg-dev libopenjp2-7-dev libpoppler-glib-dev
libraw-dev librsvg2-bin librsvg2-dev libsdl2-dev libspiro-dev
libswscale-dev libtiff-dev libumfpack5 libv4l-dev
```

```
libwebkit2gtk-4.0-dev libwebp-dev luajit python-gi-dev
python3-venv python3-wheel ruby w3m xsltproc valac libgtk-3-bin
libomp-dev cmake gobject-introspection libgirepository1.0-dev
glib-networking libappstream-glib-dev && apt upgrade
```

Il est bon de rappeler qu'il sera probablement nécessaire de refaire cette étape à chaque mise à jour majeure de **Gimp et de vérifier les nouvelles dépendances**.

Compilation du Code Source

Définir des variables d'environnement

Pour garder intact notre installation Gimp intact et ne rien casser par maladresse dans notre distro nous allons créer quelque **variables d'environnement** pour installer les sources Gimp dans notre **Bureau** sous le nom de **Gimp-Builder .**

```
export GIMP_PREFIX=$(pwd)
export GIO_MODULE_DIR=/usr/lib/x86_64-linux-gnu/gio/modules/
export PATH="$GIMP_PREFIX/bin:$PATH"
export
PKG_CONFIG_PATH="$GIMP_PREFIX/share/pkgconfig:$GIMP_PREFIX/lib/pkgcon
fig$PKG_CONFIG_PATH"
export
XDG_DATA_DIRS="${XDG_DATA_DIRS:+$XDG_DATA_DIRS:}$GIMP_PREFIX/share:/u
sr/local/share:/usr/share"
export LD_LIBRARY_PATH="$PREFIX/lib:$LD_LIBRARY_PATH"
export ACLOCAL_FLAGS="-I $GIMP_PREFIX/share/aclocal $ACLOCAL_FLAGS"
export
GI_TYPELIB_PATH="$GIMP_PREFIX/lib/girepository-1.0:$GI_TYPELIB_PATH"
```

De plus si vous êtes sous une dérivative de debian comme ubuntu il vous faudra faire ses **variables d'environnement additionnels**.

```
export arch="$(dpkg-architecture -qDEB_HOST_MULTIARCH 2> /dev/null)"
export
PKG_CONFIG_PATH="$GIMP_PREFIX/lib/$arch/pkgconfig:$PKG_CONFIG_PATH"
export LD_LIBRARY_PATH="$GIMP_PREFIX/lib/$arch:$LD_LIBRARY_PATH"
export
GI_TYPELIB_PATH="$GIMP_PREFIX/lib/$arch/girepository-1.0:$GI_TYPELIB_
PATH"
```

Télécharger le code source depuis Git

Pour télécharger les codes sources faites les manipulation suivante :

```
mkdir -R $GIMP_PREFIX/src && cd $GIMP_PREFIX/src
git clone https://gitlab.gnome.org/GNOME/babl.git
git clone https://gitlab.gnome.org/GNOME/gegl.git
git clone https://gitlab.gnome.org/GNOME/gimp.git
```

Compiler le code source

Puis exécuter ces commandes à la racine du du dossier gimp pour utiliser installer gimp de la branche **dev gimp-2-99**.

```
cd $GIMP_PREFIX/src/babl && meson _build --prefix=$GIMP_PREFIX
--buildtype=release -Db_lto=true && cd _build && ninja && ninja
install
cd $GIMP_PREFIX/src/gegl && meson _build --prefix=$GIMP_PREFIX
--buildtype=release -Db_lto=true && cd _build && ninja && ninja
install
cd $GIMP_PREFIX/src/gimp && meson _build --prefix=$GIMP_PREFIX
--buildtype=release -Dpython=enabled && cd _build && ninja && ninja
install
cd $GIMP_PREFIX/
```

Lancer Gimp

Puis lancer gimp avec la commande **$GIMP_PREFIX/bin/gimp-2.99**, vous pouvez remplacer **2.99** par la version de gimp que vous voulez compiler…

Problème rencontrez

- Principalement des erreurs dans la documentation officielle avec des oublis de mots clefs ou un changement de nom.
- Les dépendances non mises à jour, qui m'a obligé à traquer les packages grâce à stackoverflow et des Forum linux.

Gimp Core Development & Resource Development

Gimp Core Development, fait référence à une contribution au développement de **Gimp** lui-même comme des patches ou des fonctionnalités directement implémentés dans la prochaine parution de Gimp.

Gimp Resource Development, fait référence au développement plugins pour gimp ou de ressources utilisant gimp comme environnement (faire appelle à l'api ou autre ressources de gimp), ou contribuer aux développement de l'environnement de gimp lui même (apis, librairies, …).