Trying to install gimp-git in archlinux using:

yay -Sdd gimp-git --mflags "DISABLE\_MESON=0 --nocheck"

I think it is falling back yo gitlab which is very very slow in downloading. What options do I have?

When installing gimp-git from the AUR and it's falling back to GitLab with slow download speeds, you have several options to work around this:

## 1. Use a Different AUR Helper or Manual Build

Try building manually or with a different approach:

bash

*# Clone the AUR package manually*

git clone https://aur.archlinux.org/gimp-git.git

cd gimp-git

*# Edit PKGBUILD if needed (see option 3)*

makepkg -si --nocheck

## 2. Configure Git to Use a Faster Mirror

You can configure git to use GitHub instead of GitLab for GIMP:

bash

*# Add this to your ~/.gitconfig*

git config --global url."https://github.com/GNOME/gimp.git".insteadOf "https://gitlab.gnome.org/GNOME/gimp.git"

## 3. Modify the PKGBUILD Source

Edit the PKGBUILD file to use GitHub instead:

bash

*# After cloning the AUR package*

cd gimp-git

*# Edit PKGBUILD and change the source URL from GitLab to GitHub*

sed -i 's|https://gitlab.gnome.org/GNOME/gimp.git|https://github.com/GNOME/gimp.git|' PKGBUILD

## 4. Use a VPN or Different Network

Sometimes the issue is regional - try:

* Using a VPN with servers in different locations
* Switching to a different network connection
* Using mobile data temporarily

## 5. Download Source Separately

Pre-download the GIMP source:

bash

*# Clone GIMP source manually from GitHub (usually faster)*

git clone https://github.com/GNOME/gimp.git

*# Then modify the PKGBUILD to use your local copy*

## 6. Alternative: Use gimp-devel

Consider using gimp-devel instead of gimp-git if available, as it might use different sources or be pre-built.

The most reliable solution is usually option 3 - modifying the PKGBUILD to use GitHub instead of GitLab, as GitHub typically has better CDN coverage and faster speeds globally.

D

How to use makepkg -ai ignoring dependencies like babl and gegl?

To use makepkg while ignoring specific dependencies like babl and gegl, you have several options:

## 1. Skip Dependency Checks Entirely

bash

makepkg -si --nodeps

This ignores ALL dependencies, which might be too aggressive.

## 2. Modify the PKGBUILD (Recommended)

Edit the PKGBUILD file to remove or comment out specific dependencies:

bash

*# Edit the PKGBUILD*

nano PKGBUILD

*# Find the depends=() or makedepends=() arrays and remove babl and gegl*

*# For example, change:*

depends=('babl' 'gegl' 'other-deps')

*# To:*

depends=('other-deps')

*# Or comment them out:*

depends=('babl' *# Remove this*

'gegl' *# Remove this*

'other-deps')

## 3. Use --assume-installed

Tell pacman to assume certain packages are already installed:

bash

makepkg -si --asdeps --assume-installed babl,gegl

## 4. Temporarily "Satisfy" Dependencies

Create dummy packages to satisfy the dependency check:

bash

*# Create a temporary directory*

mkdir temp-deps && cd temp-deps

*# Create a dummy PKGBUILD for babl*

cat > PKGBUILD << 'EOF'

pkgname=babl

pkgver=999

pkgrel=1

arch=('any')

provides=('babl')

conflicts=('babl')

EOF

makepkg -si --nodeps

*# Repeat for gegl if needed*

## 5. Use pacman's --assume-installed During Install

bash

*# Build the package*

makepkg --nocheck

*# Install with assumption that deps exist*

sudo pacman -U \*.pkg.tar.xz --assume-installed babl,gegl

## Most Practical Approach

The cleanest method is usually **option 2** - edit the PKGBUILD to remove the problematic dependencies:

bash

cd gimp-git

cp PKGBUILD PKGBUILD.backup *# Make a backup*

sed -i '/babl/d; /gegl/d' PKGBUILD *# Remove lines containing babl or gegl*

makepkg -si --nocheck

**Warning**: Removing dependencies might cause build failures or runtime issues if those libraries are actually required. Make sure you understand why you want to skip them and that you have compatible versions installed through other means.

D

but then after cloning, makepkg -si again starts falling back to gitlab download

You're right! Even after cloning the AUR package, makepkg still needs to download the actual GIMP source code, and it's using the GitLab URL specified in the PKGBUILD. Here's how to fix this:

## Solution 1: Edit the PKGBUILD Source URL

After cloning the AUR package, modify the source to use GitHub:

bash

git clone https://aur.archlinux.org/gimp-git.git

cd gimp-git

*# Edit PKGBUILD to use GitHub instead of GitLab*

sed -i 's|git+https://gitlab.gnome.org/GNOME/gimp.git|git+https://github.com/GNOME/gimp.git|g' PKGBUILD

*# Also update any other GitLab URLs in the source array*

sed -i 's|gitlab.gnome.org|github.com|g' PKGBUILD

sed -i 's|GNOME/gimp\.git|GNOME/gimp.git|g' PKGBUILD

*# Now build*

makepkg -si --nocheck

## Solution 2: Pre-download and Use Local Source

Download the source manually and modify PKGBUILD to use it:

bash

*# Clone GIMP source from GitHub (faster)*

git clone https://github.com/GNOME/gimp.git gimp-source

*# Clone AUR package*

git clone https://aur.archlinux.org/gimp-git.git

cd gimp-git

*# Edit PKGBUILD to use local source*

*# Change the source line from:*

*# source=("git+https://gitlab.gnome.org/GNOME/gimp.git")*

*# To use your local copy:*

sed -i 's|source=(.\*|source=("gimp::git+file://'$(pwd)'/../gimp-source")|' PKGBUILD

makepkg -si --nocheck

## Solution 3: Override Git URL Globally

Set git to redirect GitLab URLs to GitHub:

bash

git config --global url."https://github.com/GNOME/".insteadOf "https://gitlab.gnome.org/GNOME/"

*# Then proceed normally*

cd gimp-git

makepkg -si --nocheck

## Solution 4: Download Source to Build Directory

bash

cd gimp-git

*# Create src directory and clone there*

mkdir -p src

cd src

git clone https://github.com/GNOME/gimp.git

cd ..

*# Modify PKGBUILD to skip the download step*

sed -i '/git+https:\/\/gitlab/d' PKGBUILD

sed -i 's|cd "$pkgname"|cd "src/gimp"|g' PKGBUILD

makepkg -si --nocheck --skipinteg

## Recommended Approach

**Solution 1** is usually the cleanest - just edit the PKGBUILD to use GitHub instead of GitLab. The GitHub mirror is typically much faster and more reliable for most users.

After editing, you can verify the change worked by checking:

bash

grep -n "source=" PKGBUILD

This should show GitHub URLs instead of GitLab ones.