

Fight Linux fragmentation with Flatpak

Jiří Janoušek
[@fenryxo](#)

<https://dl.tiliado.eu/devconf2019/>

CC-BY-SA 4.0

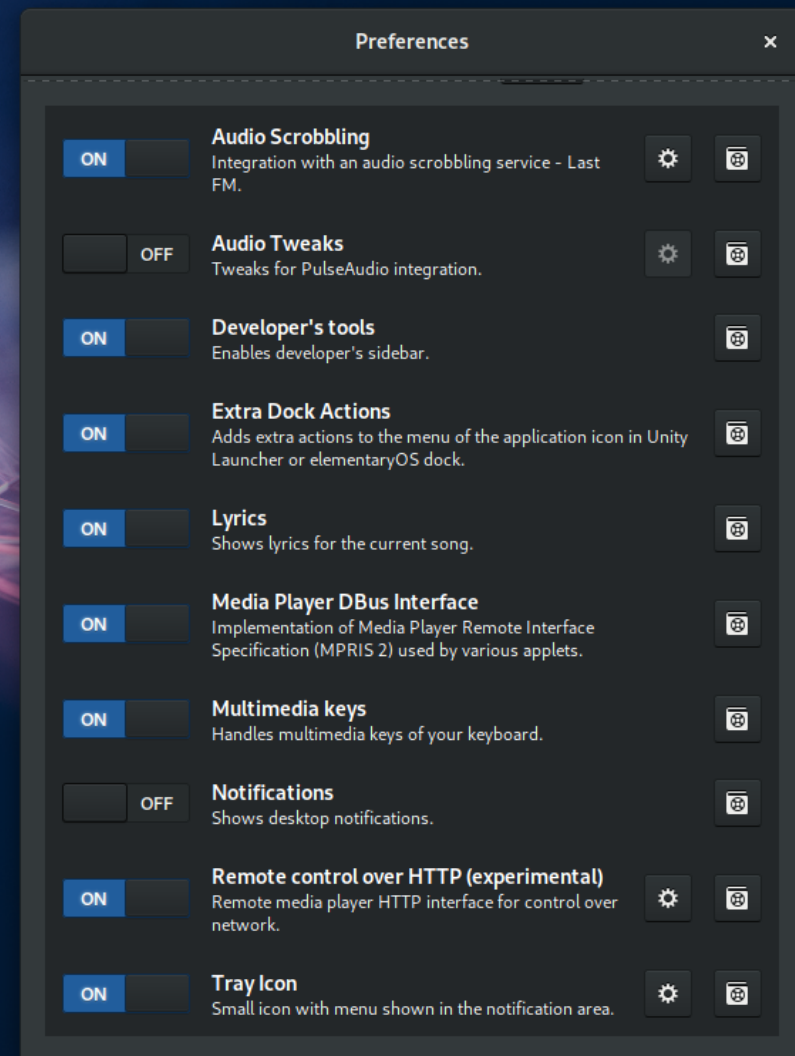
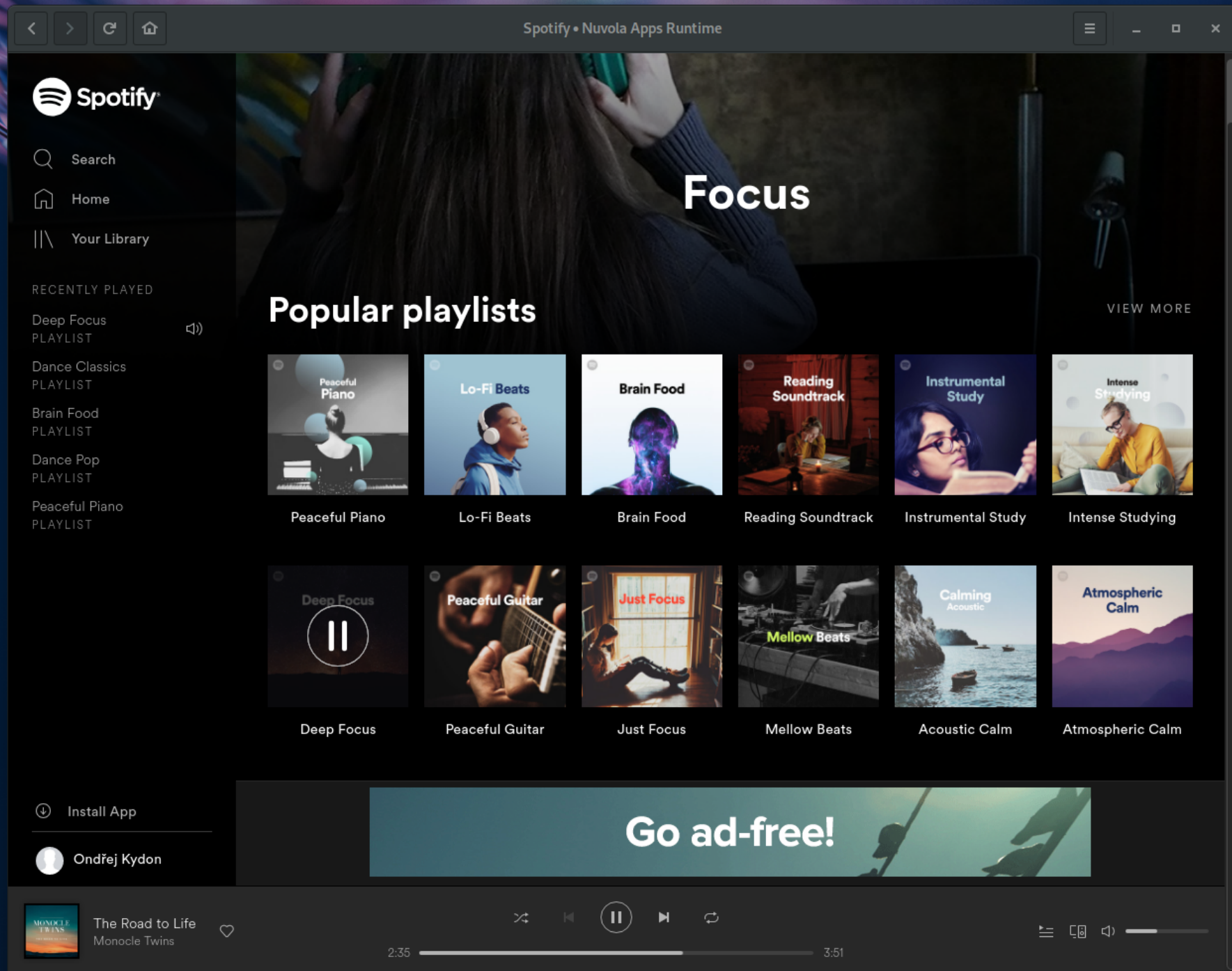
What's Flatpak?

“ Flatpak is a next-generation technology for building and distributing desktop applications on Linux. <https://flatpak.org> ”

- **Applications** as in an app store (Android, iOS)
- **Sandbox** (gradually improved)
 - You trust the developer enough to run the app
 - But not to let it read passwords and private keys
- **Social aspect:** Shorter distance between developers and users
 - Apps directly from developers, usually hosted on Flathub

Nuvola Apps Project

- Integration of **web-based music streaming services** (Spotify and 28 others) with Linux desktop to improve user experience
- Might embrace Progressive Web Apps in future
- Homepage: <https://nuvola.tiliado.eu>
- Source: <https://github.com/tiliado>
- **Technologies:**
 - GTK+ (GUI), Chromium Embedded Framework (web engine)
 - Vala (core), Python (tools), JavaScript (integration scripts)



Fragmentation of Linux

- Independent software vendors cannot target standard Linux

Fragmentation of Linux

- Independent software vendors cannot target standard Linux
- **Various distributions, versions of libraries, package formats**
 - Typically one package per distribution release
 - Packages change names, split, merge, ...
 - Maintenance burden

How Flatpak Helps

~~Various distributions, package formats, versions of libraries~~

- **Cross-distribution** Flatpak packages
- Apps running **in a sandbox**, shielded from host libraries
- **You choose the runtime** to provide `/usr` for your app
- Your app uses **the same runtime** on every distribution
- **Forward compatibility**: no need to rebuild flatpaks for a new distribution release

Nuvola and **Vala** Issues

- Nuvola core written in **Vala**
 - Nice C#/Java-like syntax
 - Vala code translated to C, then compiled
 - **VAPI (Vala API)** describes integration with C/GObject libraries
- Every release fixes bugs and improves VAPIs :-)
 - But sometimes in an **incompatible** way :-(
 - `#if VALA_0_40 ... #elif ... #else ... #endif` nightmare
 - Local copies of VAPI files (outdated, without new bug fixes)

How Flatpak Helps

~~Various distributions, package formats, versions of libraries~~

- I can target **Vala from SDK** or build it **from source with bug fixes**
- I dropped conditional compilation directives and local VAPI files
- I started refactoring, extended unit tests and run them with Valgrind
- As a result, I reported a few Vala bugs and made merge requests
- I **test with Vala from git master** (CircleCI) and report regressions

Fragmentation of Linux

- Independent software vendors cannot target standard Linux
- Various distributions, versions of libraries, package formats
- **Bug fixes not always backported**
 - Each distribution has a unique set of bugs
 - Not enough resources to fix them all
 - App developers can:
 - Find workaround
 - Suggest upgrading to the next distribution release

How **Flatpak** helps

~~Bug fixes not always backported~~

- **Full control** over dependencies
- Add **patches to fix bugs** or to customize modules
- **Override** libraries from runtime if necessary

Fragmentation of Linux

- Independent software vendors cannot target standard Linux
- Various distributions, versions of libraries, package formats
- Bug fixes not always backported
- **Some libraries or optional features are missing**
 - You are limited to libraries available in all distributions
 - You may need to sacrifice quality for compatibility
 - You may need to disable optional features depending on distro

How **Flatpak** helps

~~Some libraries or optional features are missing~~

- **Full control** over dependencies
- You can **bundle extra libraries**
- Use the best one for the given task
- Enable **all features** of your app

Nuvola and **WebKitGTK+**

- **WebKitGTK+:** library to embed web rendering engine into GTK+ applications; used by GNOME Web, Devhelp, GNOME Help, ...
- **Rapid** development of web technologies
 - but WebKitGTK in distributions **upgraded slowly**
- Desire to **switch from Flash plugin** to pure HTML5 Audio playback
 - but experimental **Media Source Extension** feature **disabled**
- **With Flatpak:** Always **up-to-date WebKitGTK with MSE enabled**

Nuvola switches to Chromium

- **WebKitGTK+**
 - **MSE** still not good enough (Google Play Music didn't work)
 - **Flash** support also [buggy](#)
 - No support for **EME/Widevine** (Spotify and Amazon Music)
- **Chromium Embedded Framework**
 - Not available in distributions as a library (no stable releases!)
 - **MSE, Widevine** and PAPI Flash supported
 - Most Nuvola apps no longer use Flash :-)

Fragmentation of Linux

- Independent software vendors cannot target standard Linux
- Various distributions, versions of libraries, package formats
- Bug fixes not always backported
- Some libraries or optional features are missing
- **Common approach: Target old distribution (RHEL, Ubuntu LTS)**
 - Workarounds for bugs, *TODO: Drop this hack in 2 years...*
 - Disconnected from platform development
 - No incentive to report and fix platform bugs

How Flatpak helps

~~Target old distribution (RHEL, Ubuntu LTS)~~









- Use the **latest stable** library stack
- Benefit from platform development:
 - **New features and bug fixes**
- Help with platform development:
 - **Report bugs** and then apply an upstream patch
 - **Fix bugs** and send patches upstream

Gentle Introduction to Flatpak SDKs, Platforms, Portals

Flatpak **Platforms**

- **Platform:** Basic libraries and Unix tools
 - Shared to save disk space and memory
 - Quality & security assurance (crypto, codecs, ...)
- **Mounted** as `/usr` in sandbox
- **Run-time dependency** of apps, installed automatically
- **Common platforms:** Freedesktop, GNOME, KDE
- **Platforms based on distributions:** Fedora, Debian








Demo 1: **org.gnome.Platform//3.30**

```
 $ flatpak remote-add --if-not-exists flathub \
    https://flathub.org/repo/flathub.flatpakrepo
 $ flatpak install flathub org.gnome.Platform//3.30
 $ flatpak run org.gnome.Platform//3.30
 $ cat /.flatpak-info
... Info about the current flatpak
 $ ls /usr /usr/bin /usr/lib/x86_64-linux-gnu
... Basic libraries and Unix tools
 $ ls /usr/include
... Almost empty
 $ gcc --version
bash: gcc: command not found
 $ python3 --version
Python 3.7.0
```

Flatpak **SDKs**

- **SDK = Platform** + development tools
 - C/C++ header files, pkg-config files
 - gcc, valac, gdb, valgrind, ...
- **SDK Extensions:** Go, Rust, Java, Mono
- **Common SDKs:** Freedesktop, GNOME, KDE
- **Build-time dependency** of apps
- **Run-time dependency** of apps for development (GNOME Builder)
- **Mounted** as `/usr` in sandbox during build

Demo 2: **org.gnome.Sdk//3.30**




```
 $ flatpak remote-add --if-not-exists flathub \
    https://flathub.org/repo/flathub.flatpakrepo
 $ flatpak install flathub org.gnome.Sdk//3.30
 $ flatpak run org.gnome.Sdk//3.30
 $ cat /.flatpak-info
... Info about the current flatpak
 $ gcc --version; valac --version; valgrind --version
gcc (GCC) 8.2.0
Vala 0.42.3
valgrind-3.13.0 # Broken: GNOME/gnome-build-meta#116
 $ ls /usr/include
... Lots of stuff
 $ pkg-config --cflags --libs gtk+-3.0
-I/usr/include/gtk-3.0 -I/usr/include/pango-1.0 ...
```

Permissions to Weaken Sandbox

- **Explicit permissions** must be set when sandbox is created
- **Filesystem access:** host, \$HOME, individual paths
- **Devices:** DRI, Bluetooth, ...; "all devices"
- **Sockets:** Xorg, Wayland, DBus (filtered), PulseAudio
- **Network** access, development API, ...
- **Long-term goal:** get rid of dangerous permissions

Demo 3: **helloworld.py**

- <https://dl.tiliado.eu/devconf2019/helloworld.py>

```
 $ flatpak run org.gnome.Platform//3.30
 $ nano helloworld.py
 $ python3 helloworld.py
Unable to init server: Could not connect: Connection refused
Unable to init server: Could not connect: Connection refused
Segmentation fault (core dumped)
```

```
 $ flatpak run --socket=x11 --socket=wayland \
    org.gnome.Platform//3.30
 $ python3 helloworld.py
... It works now!
```


Long Term Goal: **Portals**

- **Trusted DBus services** running on host, called from sandbox
- **Actions confirmed by user**
 - Opening/saving files from/on host
 - Opening URIs
 - Printing
 - Taking screenshots
 - ...
- Outside Flatpak - **screen sharing** (Wayland); **Snap** packages

Demo 4: Screenshot Portal via DBus

- <https://dl.tiliado.eu/devconf2019/screenshotlib.py>

```
 $ flatpak run --socket=x11 --socket=wayland \
    org.gnome.Platform//3.30
```

```
 $ nano screenshotlib.py
```

```
 $ python3 -i screenshotlib.py
```

```
>>> service = ScreenshotService()
>>> screenshot = service.take_screenshot()
>>> path1 = screenshot.get_result().get_path()
>>> path1
'/run/user/1000/doc/32da9cd2/Screenshot - 1.png'
```

Demo 5: File Chooser Portal via GTK+

```
>>> # Follow-up from demo 4
>>> from gi.repository import Gtk
>>> chooser = Gtk.FileChooserNative.new("Save screenshot", None,
...     Gtk.FileChooserAction.SAVE, "Save", "Cancel")
>>> chooser.run()
-3
>>> path2 = chooser.get_filename(); path2
'/run/user/1000/doc/7ecd1fc0/lo1.png'
>>> with open(path1, 'rb') as f1, open(path2, 'wb') as f2:
...     f2.write(f1.read())
...
58938
>>> import os
>>> os.unlink(path1)
```

Flatpak **Builder**

- **Build flatpaks** from manifests (JSON/YAML)
- **Source:** directory; archive; git, bzd, svn; patch files
- **Build system:** configure & make, autotools, cmake, meson; custom
- Sources downloaded and verified (checksums)
- Building with limited filesystem and no network access
- **Incremental builds** - each module cached

Demo 6: eu.tiliado.Hello.yaml

- <https://dl.tiliado.eu/devconf2019/helloworld.py>
- <https://dl.tiliado.eu/devconf2019/eu.tiliado.Hello.yaml>

```
 $ flatpak install flathub \
    org.freedesktop.Platform//18.08 org.freedesktop.Sdk//18.08
 $ flatpak-builder --install --user hello eu.tiliado.Hello.yaml
 $ ls .flatpak-builder # State dir: downloads, cache, ...
 $ flatpak run eu.tiliado.Hello
 $ flatpak run --command=bash eu.tiliado.Hello
 $ cat /.flatpak-info; cat /app/manifest.json
 $ ls /app /app/bin /app/share/icons \
    /app/lib/python3.7/site-packages/gi
 $ hello
```

Development

Inside Sandbox

Develop **Inside Sandbox**

- Develop in the **same environment** users will your run app in
 - Easier **reproducibility** of bugs
- **Bash session in the sandbox**
 - `flatpak run --devel --command=bash --filesystem=~/projects ...`
- **GNOME Builder** (available as a Flatpak)
 - Can build apps in sandbox and create bundles
- **KDevelop**
 - [Flatpak support in KDevelop – Jan Grulich](#)

Demo 7: GNOME Builder

- <https://dl.tiliado.eu/devconf2019/screenshot-demo-app.tar.xz>
- Install GNOME Builder
 - Fedora: `dnf install gnome-builder`
 - Flathub: `flatpak install flathub org.gnome.Builder`
- Open the project from the screenshot-demo-app directory.
- Build.
- Run.

Build status window for screenshot-demo-app:

- Branch: unversioned
- Build Profile: eu.tiliado.Screenshot.json
- Runtime: org.gnome.Platform 3.30

Build status:

- Last build: Success
- Errors: 0
- Warnings: 0

Buttons: Build, Rebuild, Clean, Export Bundle

Terminal output:

```
pp --filesystem=/home/fenryxo/.cache/gnome-builder/projects/screenshot-demo-app/builds/eu.tiliado.Screenshot.json-flatpak-org.gnome.Platform-x86_64-3.30-unversioned --env=V=1 '--env=CFLAGS=-O2 -g' '--env=CXXFLAGS=-O2 -g' --env=NOCONFIGURE=1 /home/fenryxo/.cache/gnome-builder/projects/screenshot-demo-app/flatpak/staging/x86_64-unversioned meson /home/fenryxo/devconf2019/screenshot-demo-app . --prefix /app --libdir=lib
```

The Meson build system

Version: 0.48.2

Source dir: /home/fenryxo/devconf2019/screenshot-demo-app

Build dir: /home/fenryxo/.cache/gnome-builder/projects/screenshot-demo-app/builds/eu.tiliado.Screenshot.json-flatpak-org.gnome.Platform-x86_64-3.30-unversioned

Build type: native build

Project name: tiliado-screenshot

Project version: 0.1.0

Build machine cpu family: x86_64

Build machine cpu: x86_64

Program desktop-file-validate found: YES (/usr/bin/desktop-file-validate)

Program appstream-util found: YES (/usr/bin/appstream-util)

Program glib-compile-schemas found: YES (/usr/bin/glib-compile-schemas)

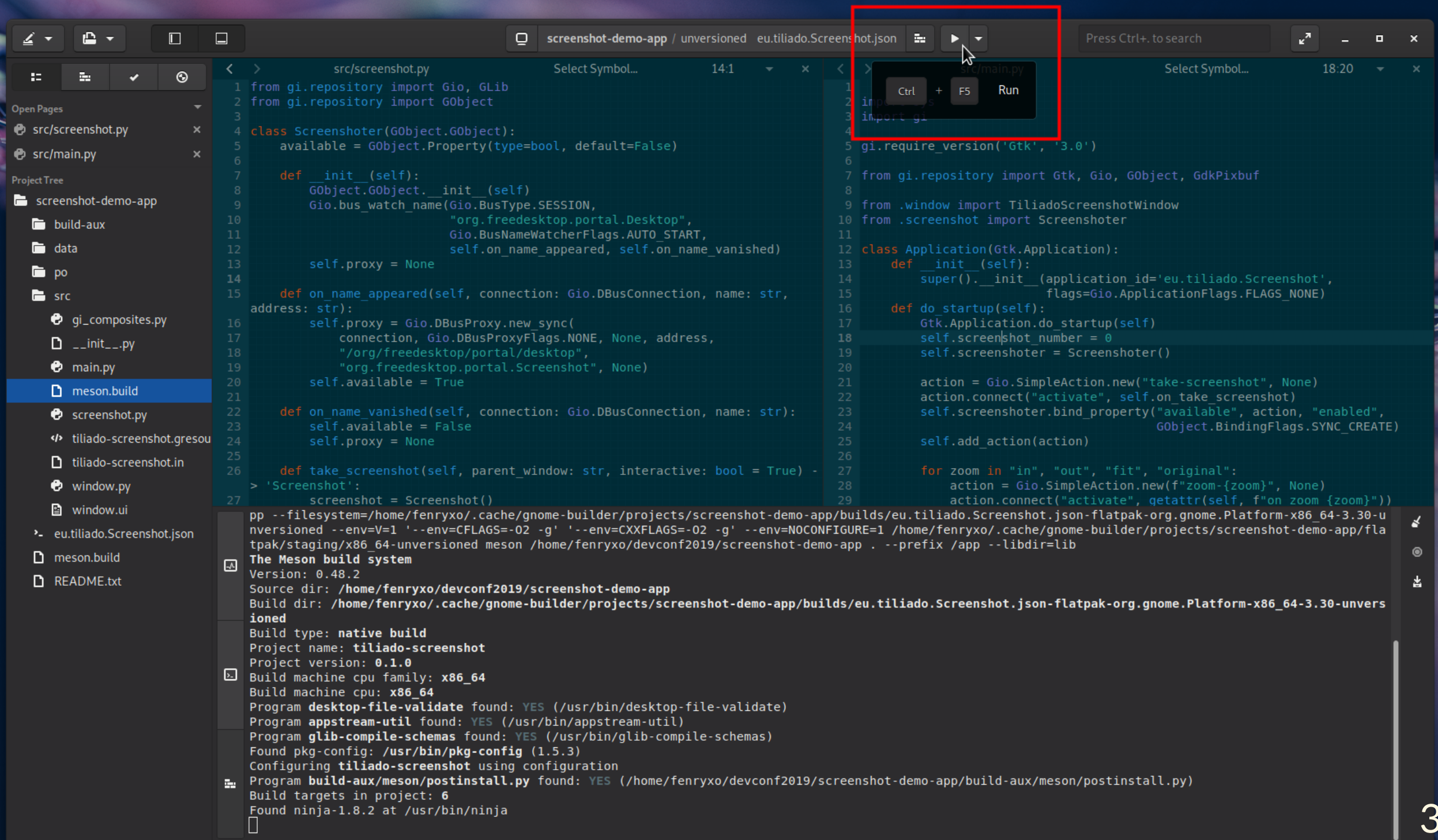
Found pkg-config: /usr/bin/pkg-config (1.5.3)

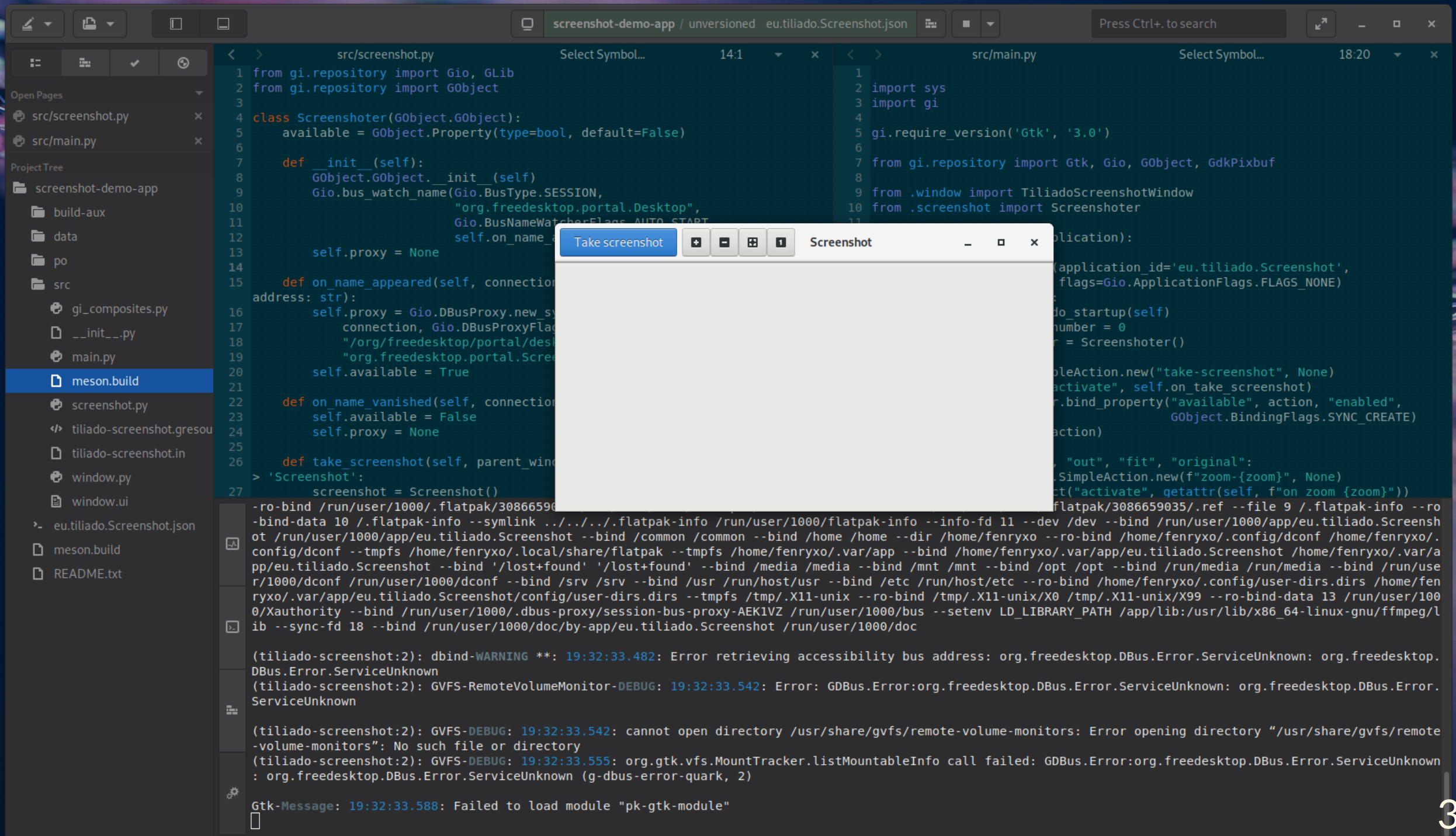
Configuring tiliado-screenshot using configuration

Program build-aux/meson/postinstall.py found: YES (/home/fenryxo/devconf2019/screenshot-demo-app/build-aux/meson/postinstall.py)

Build targets in project: 6







Found ninja-1.8.2 at /usr/bin/ninja












Nuvola CDK: **Core Developer Kit**

- [Nuvola CDK](#): Flatpak with **all deps for Nuvola** but without Nuvola
- **Development of Nuvola** in the same sandbox environment the app will run in
- **Quick set-up** everywhere, especially in **Boxes/VirtualBox**

```
 $ flatpak install nuvola eu.tiliado.NuvolaCdk
 $ flatpak run -d --filesystem=~/projects eu.tiliado.NuvolaCdk
 $ cd ~/projects/nuvolaruntime
 $ . setup_nuvolacdk.sh
 $ rebuild
 $ run-app nuvola-app-nuvola-demo-player
```

Nuvola ADK: **App Developer Kit**

- [Nuvola ADK](#) flatpak extends Nuvola CDK with
 - **prebuilt Nuvola Runtime**
 - **Nuvola SDK** (a few helper scripts in Python/Bash)
- Development of **web app integration scripts** in **JavaScript** ([tutorial](#))

```
 $ flatpak install nuvola eu.tiliado.NuvolaAdk
 $ flatpak run --filesystem=~/projects eu.tiliado.NuvolaAdk
 $ cd ~/projects
 $ nuvolasdk new-project ...
 Edit metadata.in.json integrate.js in your favorite editor...
 $ nuvolasdk check-project; ./configure; make all
 $ nuvolaruntime --debug
```

Production **Nuvola Flatpaks**

- **One Flatpak per script**
 - eu.tiliado.**NuvolaAppDeezer**, **NuvolaAppGoogleCalendar**, ...
 - Can run independently side-by-side
 - metadata.json, integrate.js, icons, desktop file, AppStream
- **Fast builds** - Nuvola & deps already prebuilt in a base Flatpak
- **Bug fixes** - Flatpaks rebuilt & published as soon as possible
- **Circle CI** - `nuvolasdk check-project` in Nuvola ADK

Drawbacks of Flatpak Packaging

- Maintenance of **bundled libraries**
 - Script to update manifest with the latest versions
 - Version information from <https://release-monitoring.org/api>
- **Missing functionality** in sandbox - talk to Flatpak devs
- Poor **command line experience** ([ticket](#)):
 - `flatpak run --command=nuvolactl eu.tiliado.Nuvola track-info`
- **New limiting factor** on host OS - version of Flatpak, Portals, ...

Conclusion

- Without Flatpak, Nuvola would be stuck with Flash plugin for audio playback or discontinued.
- Flatpak provides:
 - Stable platform to optimize for and rely on.
 - Cross-distribution app delivery.
 - Full control over dependencies.
 - Full control over quality.
 - Direct interaction with users.