Guide technique POP Switcher

- Version documentée : v2.28
- Application : POP Switcher, local AI orchestrator
- Ancien nom : Glob, puis SwitchAI
- Dernière mise à jour du guide : 7 juin 2026

Sommaire

- Démarrage rapide : sections 1 à 5.
- Guide des rôles et des UI : sections 6 à 16.
- Host Bridge, mémoire et architecture : sections 17 à 20.
- Modèles, permissions, performance et dépannage : sections 21 à 24.
- Notes développeur et stratégie de maintenance : sections 25 à 27.

Public visé

Ce guide est écrit pour deux profils à la fois. D'abord l'utilisateur qui veut installer ses dépendances, choisir un modèle, lancer une UI et retrouver ses fichiers. Ensuite le développeur qui doit comprendre l'architecture, les rôles, les chemins, le Host Bridge, The One et les points de vigilance avant de modifier le code.

1. Résumé court

POP Switcher est une application macOS conçue pour rendre l'IA locale simple, visuelle et accessible. Son objectif est de permettre à un utilisateur de télécharger ses modèles, installer les dépendances nécessaires, lancer une interface adaptée et retrouver facilement ses créations, sans devoir passer son temps dans le Terminal.

L'idée centrale est simple : chaque usage de l'IA devient un rôle. Un rôle peut être Chat, Code, Réflexion, Vision, Image, Vidéo, Son, Retouche Image, The One ou un rôle personnalisé. Chaque rôle peut avoir son propre modèle, son interface, ses réglages et ses dépendances.

POP Switcher ne cherche pas seulement à lancer un gros modèle unique. L'app permet aussi de faire travailler plusieurs petits modèles spécialisés ensemble. C'est le rôle de The One : découper une demande complexe en étapes, appeler les bons rôles, garder le suivi du projet et aider l'utilisateur à construire quelque chose sur la durée.

POP Switcher peut aussi booster des modèles locaux en leur ajoutant des outils : recherche internet pour le rôle Réflexion, workspace texte sécurisé, génération d'images, retouche, audio, vidéo ou code selon les rôles installés. La conception reste ouverte : l'utilisateur peut ajouter des rôles, des dépendances personnalisées, des LoRA, des modèles spécialisés et faire évoluer son installation sans repartir de zéro.

En résumé : POP Switcher est un orchestrateur local d'IA, pensé pour rendre les modèles locaux utilisables au quotidien, compréhensibles pour un débutant et suffisamment structurés pour qu'un développeur puisse maintenir ou étendre l'app.

2. Vocabulaire essentiel

Rôle : fonction d'IA dans POP Switcher, par exemple `chat`, `image`, `audio` ou `theOne`.

Modèle : fichier ou dossier de modèle IA, souvent en GGUF, Diffusers ou autre format local.

Runtime : programme qui lance réellement le modèle, par exemple llama.cpp, stable-diffusion.cpp, ComfyUI ou acestep.cpp.

UI : interface dédiée au rôle. Exemple : UI Chat, UI Code, UI Image, UI Vidéo.

Dépendance : composant nécessaire avant de lancer un modèle. Exemple : Homebrew, Python, llama.cpp, stable-diffusion.cpp, Wan video stack, ACE-Step runtime.

WorkFlow : dossier utilisateur contenant les modèles, historiques, mémoires, codes, images, vidéos, audio et LoRA.

Host Bridge : serveur HTTP local sur `127.0.0.1`, protégé par token, permettant à switchCORE de piloter POP Switcher sans UI scripting fragile.

The One : rôle orchestrateur long terme capable de découper un projet en étapes et d'appeler d'autres rôles comme outils.

3. Guide de démarrage rapide

Étape 1 : lancer POP Switcher

Ouvre l'app `POP Switcher-v2.28.app`. Au premier lancement, l'application crée ou vérifie les dossiers de base.

Par défaut, les fichiers utilisateur sont rangés dans :

```
~/Applications/WorkFlow
```

Le dossier de modèles par défaut est :

```
~/Applications/WorkFlow/LLM
```

Ce choix évite autant que possible les demandes répétées d'autorisation macOS liées à `Documents`, `Desktop` ou `Downloads`.

Étape 2 : ouvrir l'aide système

Dans la tuile Système, clique sur le bouton `?`. Cette aide indique :

- les modèles testés et recommandés ;
- les dépendances nécessaires pour chaque modèle ;
- les liens Hugging Face ;
- les réglages recommandés quand ils sont connus ;
- les LoRA conseillées pour certains modèles.

Étape 3 : installer les dépendances

Dans Système ou Réglages > Dépendances, installe les composants nécessaires au rôle que tu veux utiliser.

Exemples :

- Chat, Code, Réflexion, The One : `llama.cpp`.
- Image : `stable-diffusion.cpp` et parfois VAE + text encoder.
- Retouche Image : `stable-diffusion.cpp`, Qwen-Image VAE et Qwen2.5-VL 7B encoder.

• Vidéo : ComfyUI / Wan video stack, VAE Wan, text encoder Wan, ffmpeg.

• Son : ACE-Step runtime, ACE-Step LM, ACE-Step encoder, ACE-Step VAE.

Homebrew et Python sont des prérequis communs utiles pour de nombreuses dépendances.

Étape 4 : télécharger le modèle

Depuis l'aide système ou les recommandations, ouvre le lien Hugging Face du modèle voulu.

Conseils pratiques :

• Pour llama.cpp et beaucoup de workflows locaux, privilégier un fichier `.gguf`.

• Pour commencer, choisir souvent une quantification `Q4_K_M` ou `Q5_K_M`.

• Plus le modèle est gros, plus il consomme de RAM et plus il sera lent.

• Ne télécharge pas un dépôt complet énorme si l'aide indique de prendre seulement une quantification précise.

• Pour les modèles vidéo Diffusers, il faut parfois télécharger un dossier complet, pas seulement un fichier.

Étape 5 : placer le modèle dans le dossier LLM

Place les modèles dans :

```
~/Applications/WorkFlow/LLM
```

Tu peux changer ce chemin dans Réglages si tu veux mettre les LLM sur un disque externe.

Étape 6 : scanner et assigner au rôle

Dans l'écran principal :

1. clique sur `Scanner` ;
2. sélectionne le modèle détecté ;
3. choisis le rôle à gauche ;
4. clique sur `Assigner au rôle`.

Étape 7 : choisir le type d'UI

Dans Commandes, choisis l'UI adaptée :

- UI Chat pour discuter simplement ;
- UI Chat+Image pour multimodal avec image ou document ;
- UI Code pour travailler sur des fichiers ;
- UI Réflexion pour web, workspace texte et chaînes autonomes ;
- UI The One pour projets longs ;
- UI Image pour générer des images ;
- UI Retouche Image pour éditer une image ;
- UI Vidéo pour générer ou animer de la vidéo ;
- UI Son pour générer de l'audio ou de la musique.

Étape 8 : lancer le runtime et ouvrir l'UI

Clique sur le bouton de lancement du runtime, puis ouvre l'UI. Certaines UI s'ouvrent automatiquement si le rôle est lancé par The One ou par le Host Bridge.

Tutoriel illustré de démarrage

Cette séquence reprend le parcours complet conseillé pour un premier test, depuis la fenêtre principale jusqu'à la vérification que le chat répond.

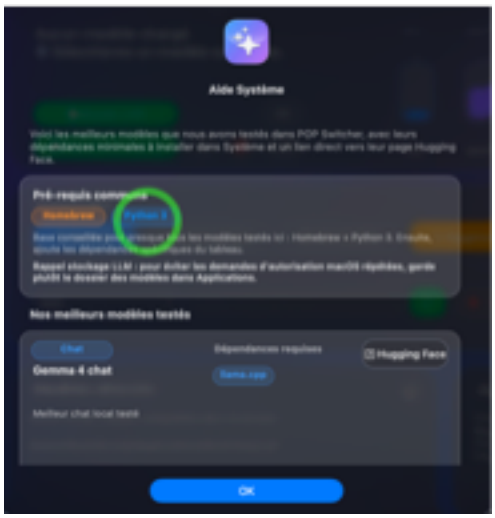
1. Ouvrir la fenêtre principale de POP Switcher.

Fenêtre principale POP Switcher

2. Ouvrir l'aide système avec le bouton `?` pour voir les modèles recommandés et les dépendances nécessaires.

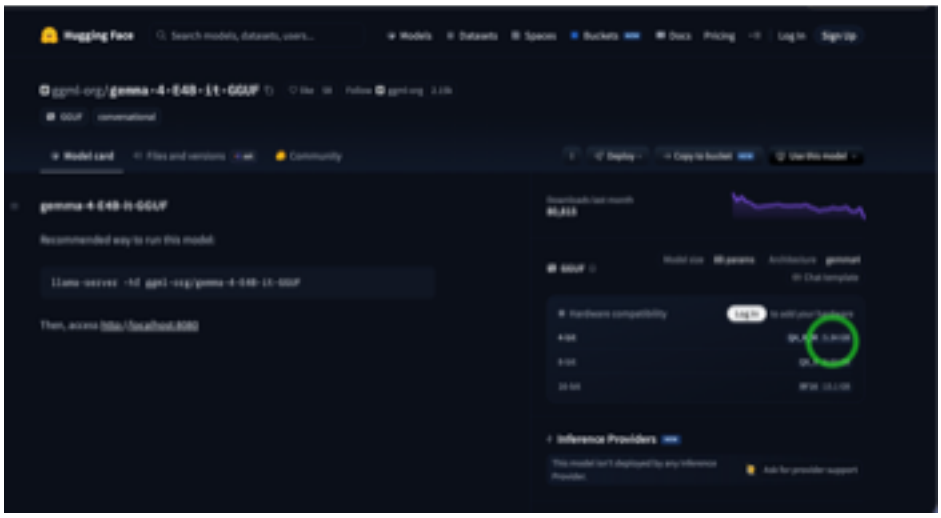
Aide système pour voir les dépendances

3. Identifier les dépendances principales demandées par le rôle ou le modèle choisi.



Dépendances requises principales

4. Ouvrir la fiche Hugging Face du modèle recommandé.



Télécharger un modèle sur Hugging Face

5. Télécharger le bon fichier modèle. Pour un GGUF, choisir en général une quantification équilibrée comme `Q4_K_M`, `Q5_K_M` ou `Q3_K_M` selon la RAM disponible.

Choisir le modèle à télécharger sur Hugging Face

6. Placer le LLM dans le dossier de modèles configuré par POP Switcher, par défaut `WorkFlow/LLM`.

[Image introuvable : Placer le LLM dans le dossier]

7. Cliquer sur `Scanner` pour faire apparaître les modèles détectés.

Scanner le dossier de modèles

8. Si une dépendance manque, cliquer sur son bouton d'installation dans Système ou Réglages > Dépendances.

Installer une dépendance manquante

9. Vérifier dans le journal que l'installation progresse ou se termine correctement.

Vérifier la dépendance dans le journal

10. Quand la dépendance est installée, relancer si besoin le scan ou le runtime. Si l'installation échoue, installer d'abord les autres dépendances requises par le modèle, puis réessayer.

[Image introuvable : Dépendance installée]

11. Choisir l'UI adaptée au rôle : UI Chat, UI Image, UI Code, UI Son, UI Vidéo, UI Retouche ou autre.

Choix de l'UI

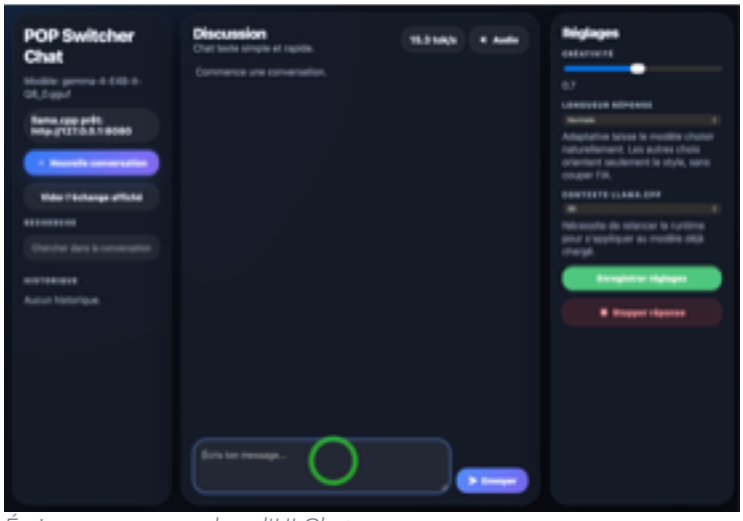
12. Lancer le modèle avec le bouton de démarrage du runtime.

Lancer le modèle

13. Ouvrir l'UI du rôle.

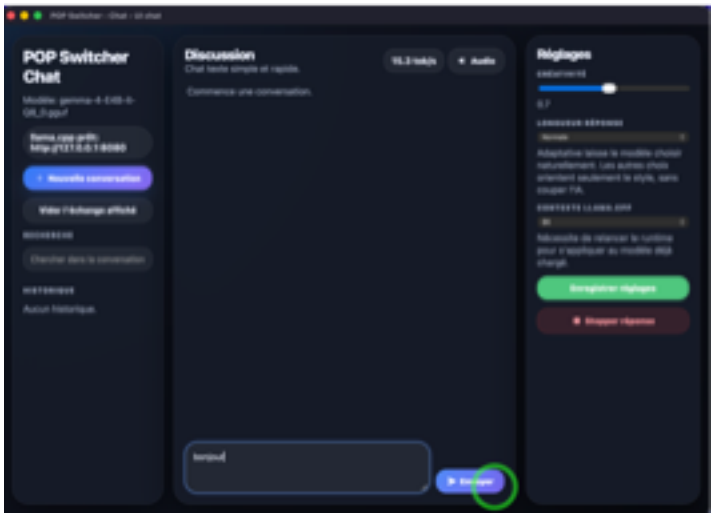
Ouvrir UI

14. Pour tester le rôle Chat, écrire un premier message simple.



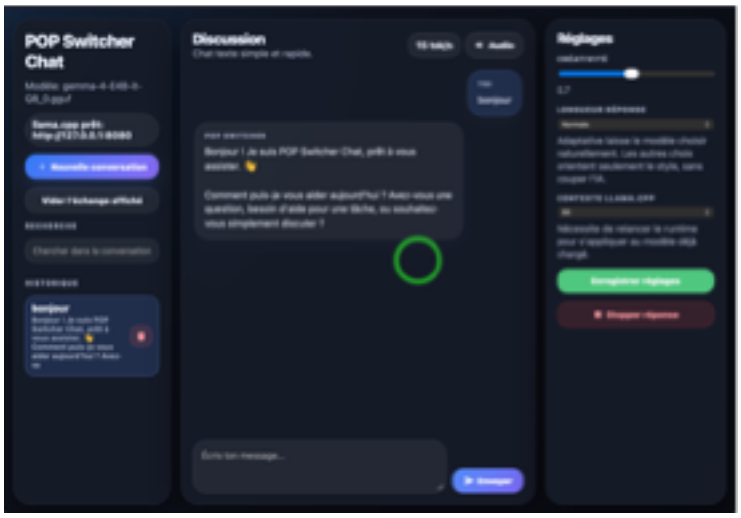
Écrire un message dans l'UI Chat

15. Envoyer le message.



Envoyer le message dans l'UI Chat

16. Vérifier que le modèle répond correctement.



Vérification de la réponse du Chat

17. Pour gagner du temps, assigner le modèle et l'UI par défaut au rôle choisi. Cela permet à POP Switcher de retrouver automatiquement la bonne combinaison.

[Image introuvable : Assigner le rôle]

18. Une fois le rôle assigné, cliquer sur un rôle rapide peut lancer ou ouvrir directement l'UI associée.

Cliquer sur un rôle pour lancer l'UI

19. Les rôles rapides servent ensuite de tableau de bord pour changer rapidement de fonction IA.

Accès rapide aux rôles

4. Organisation des dossiers

POP Switcher sépare volontairement les dépendances techniques et les fichiers utilisateur.

Dossiers techniques

Les composants internes sont dans :

```
~/Library/Application Support/POP Switcher
```

Sous-dossiers principaux :

```
~/Library/Application Support/POP Switcher/runtimes
~/Library/Application Support/POP Switcher/models
~/Library/Application Support/POP Switcher/logs
~/Library/Application Support/POP Switcher/state
~/Library/Application Support/POP Switcher/dependencies
```

Ces dossiers contiennent les runtimes, dépendances, logs, états serveur et modèles auxiliaires techniques.

Dossiers utilisateur WorkFlow

Par défaut :

```
~/Applications/WorkFlow/LLM
```

```
~/Applications/WorkFlow/Audio
~/Applications/WorkFlow/Image
~/Applications/WorkFlow/Vidéo
~/Applications/WorkFlow/Historique
~/Applications/WorkFlow/Mémoire
~/Applications/WorkFlow/Code
~/Applications/WorkFlow/LoRA/LoRA Son
~/Applications/WorkFlow/LoRA/LoRA Image
~/Applications/WorkFlow/LoRA/LoRA Vidéo
~/Applications/WorkFlow/LoRA/LoRA Retouche
```

Préférences

macOS garde encore le fichier système :

```
~/Library/Preferences/BRUNET-FLORENT-EI.GLOB.plist
```

POP Switcher le miroir aussi dans :

```
~/Applications/WorkFlow/Mémoire/Preferences/BRUNET-FLORENT-EI.GLOB.plist
```

Le nom de bundle historique reste `BRUNET-FLORENT-EI.GLOB` pour éviter de casser les préférences existantes.

5. Réglages principaux

Les réglages de POP Switcher sont organisés par onglets.

Onglet Général

Contient les réglages utilisateur les plus simples :

- dossier WorkFlow ;
- dossier des modèles LLM ;
- dossier de sortie image ;
- mode d'affichage ;
- comportement général de l'app ;
- remise à zéro des modèles ou des réglages si nécessaire.

Onglet Dépendances

Contient :

- Homebrew ;
- Python ;
- llama.cpp ;
- stable-diffusion.cpp ;
- ComfyUI / Wan stack ;
- ffmpeg ;
- ACE-Step ;
- dépendances auxiliaires comme VAE et text encoders ;
- dépendances personnalisées ajoutées par l'utilisateur.

Une dépendance personnalisée peut avoir une commande d'installation, de mise à jour et de désinstallation. Cela permet à un utilisateur avancé d'ajouter une stack locale non encore connue par l'app.

Onglet Profil

Contient :

- le profil machine ;
- les recommandations de modèles selon la machine ;
- les modèles conseillés pour chaque rôle ;
- les dépendances requises par modèle.

Onglet Avancé

Contient :

- Host Bridge local ;
- token local ;
- endpoints par rôle ;
- jobs et artefacts ;

- automatisation switchCORE ;
- mémoire Réflexion ;
- réglages de contexte par rôle ;
- options de runtime plus sensibles.

6. Les rôles intégrés

Les rôles internes stables sont :

```
chat
code
reasoning
theOne
vision
image
video
audio
imageEdit
```

Ces identifiants ne doivent pas changer, même si le nom affiché dans l'UI change.

Tableau de synthèse

```
Rôle        Usage principal                         Runtime typique
chat        Conversation simple                      llama.cpp
code        Génération/modification de code           llama.cpp
reasoning   Recherche web et chaînes autonomes        llama.cpp + outils internes
theOne      Orchestration de projets longs            llama.cpp + rôles outils
vision      Analyse image/document                    llama.cpp multimodal + mmproj si
besoin
image       Génération image                          stable-diffusion.cpp
imageEdit   Retouche image                            stable-diffusion.cpp
video       Génération vidéo                          ComfyUI / Wan / LTX
audio       Musique et son                            acestep.cpp / ACE-Step
```

7. UI Chat

L'UI Chat est volontairement simple et rapide.

Elle sert à :

• discuter avec le modèle attribué au rôle `chat` ;

• gérer un historique de conversations ;

• rechercher dans la conversation ;

• vider ou supprimer une conversation ;

• régler la créativité ;

• orienter la longueur de réponse sans couper brutalement la génération ;

• activer une lecture audio des réponses si disponible.

Le but de cette UI est de rester légère. Elle ne charge pas de gros module vision par défaut.

8. UI Chat + Image / Vision

Cette UI est prévue pour les modèles multimodaux.

Elle peut utiliser :

• un modèle compatible vision ;

• éventuellement un fichier `mmproj` selon le modèle et le backend ;

• des images ou documents glissés dans l'interface.

Elle est plus lourde que l'UI Chat simple. Elle doit être utilisée seulement si le rôle ou le modèle a réellement besoin d'entrée image ou document.

9. UI Code

L'UI Code est un workspace sécurisé pour demander à un LLM de créer, modifier et expliquer du code.

Dossier de travail

Par défaut :

```
~/Applications/WorkFlow/Code
```

Le code est organisé en projets et fichiers. L'interface affiche une arborescence pour sélectionner les

fichiers.

Fonctions principales

• créer un nouveau fichier sans popup ;

• renommer un fichier ;

• supprimer un fichier ou dossier avec confirmation dans l'interface ;

• sélectionner un ou plusieurs fichiers ;

• afficher les fichiers sélectionnés en onglets ;

• modifier l'onglet courant ;

• modifier tous les fichiers cochés ;

• poser une question rapide via le compte rendu sans modifier le code ;

• construire une app macOS `.app` quand le projet Swift est suffisamment correct ;

• préparer des exports Windows ou Linux ;

• versionner les modifications.

Versionning

Quand le versionning est actif, l'app évite d'écraser l'ancien travail. L'idée recommandée est :

```
Projet/
  v1.01/
  v1.02/
  v1.03/
```

Chaque modification peut produire un nouveau sous-dossier de version avec les fichiers modifiés.

Bonnes pratiques

Avant de demander à l'IA de coder :

1. créer un projet ou un fichier ;

2. sélectionner le ou les fichiers concernés ;

3. écrire une demande précise ;

4. utiliser `Chat rapide CR` pour poser une question sans modifier le code ;

5. utiliser `Modifier l'onglet` pour une correction ciblée ;

6. utiliser `Modifier fichiers cochés` pour une modification multi-fichiers.

Limites

Le modèle peut écrire du code invalide. POP Switcher peut lancer une construction macOS, mais il ne remplace pas Xcode. Pour les apps macOS complexes, il faudra parfois corriger les imports, le nom de l'App, les fichiers SwiftUI, les assets ou les paramètres de compilation.

10. UI Réflexion

L'UI Réflexion sert à donner plus d'autonomie au modèle.

Elle ajoute :

- recherche DuckDuckGo ;
- exploration de pages web ;
- workspace texte sécurisé ;
- chaînes autonomes ;
- mémoire web ;
- préprompt d'outils ;
- relance de chaîne si le but n'est pas atteint.

Principe d'une chaîne autonome

Quand l'utilisateur demande une tâche complexe, le modèle peut d'abord analyser la demande, puis produire une chaîne d'actions.

Exemple :

```
1. //recherche internet DUCK DUCK "POP stacker"
2. //ouvrir résultat utile
3. //synthèse
4. //création de fichier dans workspace
5. //réponse finale utilisateur
```

L'utilisateur voit :

- le but ;
- les étapes ;
- le journal d'exécution ;
- les blocages ;
- les tâches récentes.

Workspace texte

Le workspace texte est limité par sécurité à des fichiers simples :

```
.txt
.md
.json
.csv
```

L'IA ne doit pas écrire n'importe où sur le disque.

Mémoire Réflexion

La mémoire Réflexion sert surtout à garder :

- résultats web ;
- pages explorées ;
- fichiers sélectionnés ;
- résumé utile ;
- état de chaîne autonome.

Elle n'est pas conçue pour devenir une mémoire infinie. Quand elle devient trop lourde, il faut la compresser, résumer ou vider certains éléments.

11. UI Image

L'UI Image sert à créer une image à partir d'un prompt.

Dépendances typiques

- stable-diffusion.cpp ;

• VAE selon modèle ;

• text encoder selon modèle ;

• LoRA optionnels.

Modèles recommandés

Z-Image-Turbo GGUF est le modèle image testé principal.

Réglages recommandés :

```
Sampler : Euler
Steps : 8
Guidance : 1.0
```

Aide au prompt

L'UI Image contient des suggestions organisées :

• sujet ;

• style ;

• lumière ;

• composition ;

• qualité ;

• caméra ;

• négatif.

Le texte ajouté par les boutons apparaît dans le champ prompt. L'objectif est d'aider un utilisateur non expert à construire un prompt plus fiable.

Résolution

Plus la résolution est grande, plus la génération est lente et gourmande. L'app propose des formats courants : 1:1, 4:3, 16:9, 21:9, YouTube, LinkedIn, Facebook, UHD, 4K, ultrawide.

12. UI Retouche Image

L'UI Retouche Image sert à modifier une image existante.

Modèle recommandé

Qwen-Image-Edit 2511 Q3_K_M GGUF.

Dépendances :

- stable-diffusion.cpp ;
- Qwen-Image VAE ;
- Qwen2.5-VL 7B encoder.

Réglages recommandés :

```
Résolution : 768 à 1152 px
Sampler : Euler
Steps : 16
Guidance : 2.5
Force : 0.25
```

LoRA conseillée :

```
Qwen-Image-Edit-2511-Lightning 8 steps bf16
Euler, 8 steps, guidance 2.5, force 0.20
```

Modes utiles

- retouche libre ;
- supprimer un objet ;
- découpe PNG ;
- fusion de deux images ;
- édition locale avec zone sélectionnée.

Sélection et masque

L'utilisateur peut charger une image par glisser-déposer. L'interface peut aider à indiquer une zone avec rectangle, cercle ou sélection guidée. L'IA doit malgré tout prendre l'image complète en compte pour reconstruire un arrière-plan cohérent.

Conseils

Pour enlever un objet, le prompt doit être direct :

```
Remove the selected object and reconstruct the background naturally.
```

Pour une découpe PNG :

```
Remove the background, keep the main subject, transparent PNG.
```

13. UI Vidéo

L'UI Vidéo est la plus expérimentale et la plus lourde.

Elle repose principalement sur ComfyUI et des workflows Wan ou LTX.

Modes

- texte vers vidéo ;
- image vers vidéo ;
- texte + image vers vidéo ;
- animer un visuel ;
- son vers vidéo, quand le workflow existe.

Wan 2.2 TI2V-5B Turbo GGUF

Modèle testé pour image vers vidéo.

Réglages validés :

```
Résolution : 640 x 360
Durée : 4 s
FPS : 16
Profil : Stable
Sampler : Euler
Scheduler : Simple
CFG : 1.0
Force : 1.0
Motion strength : 8
Steps : 4
```

```
LatentMultiply : décoché
Nettoyage léger : décoché
CLIP-Vision : non utilisé pour ce workflow validé
```

Le modèle Turbo a mieux fonctionné en image-to-video qu'en texte seul. En texte seul, il a produit des artefacts ou des résultats psychédéliques.

Wan 2.2 TI2V-5B GGUF Squirrelae

Workflow 5B plus accessible :

```
Un seul GGUF
VAE Wan 2.2
UMT5
640 x 360
4 s
16 FPS
Euler
Scheduler Simple
CFG 3.5
Shift 8
20 à 30 steps
```

Résultat observé : fonctionnel mais qualité jugée trop faible pour être le meilleur choix.

Wan 2.2 T2V-A14B GGUF

Vrai modèle texte vers vidéo, plus ambitieux et plus lourd.

Il faut deux GGUF de même quantification :

```
HighNoise
LowNoise
```

Conseil de départ :

```
Q2_K ou Q3_K_S
640 x 360
4 s
Euler
Scheduler Simple
CFG 5.0
Shift 5
```

```
20 steps
Sans LoRA au premier test
```

Sur Mac 24 Go, ce modèle peut être trop ambitieux.

LTX2.3-10Eros GGUF

POP Switcher peut préparer ou lancer ComfyUI pour LTX, mais la génération directe LTX est volontairement protégée tant que le graphe exact n'est pas validé.

Raison : un mauvais wiring de nodes ComfyUI peut produire des vidéos absurdes ou incohérentes. Le runtime peut être préparé, mais le bouton Générer doit éviter de promettre une génération fiable tant que le workflow n'a pas été stabilisé.

Sécurité ressources

La vidéo peut prendre plusieurs heures et saturer RAM, GPU ou CPU. Il faut commencer par :

```
512 x 288 ou 640 x 360
4 secondes
4 à 8 steps si modèle turbo
20 steps maximum pour un premier test non turbo
```

14. UI Son

L'UI Son sert à générer de la musique ou du son.

Modèle recommandé

Serveurperso/ACE-Step-1.5-GGUF.

Dépendances :

- ACE-Step runtime ;
- ACE-Step LM ;
- ACE-Step text encoder ;
- ACE-Step VAE.

Réglages recommandés de départ :

```
Mode : turbo
Steps : 8
Guidance : 1.0
Shift : 3.0
Durée : 240 s par défaut dans l'UI actuelle
```

Interface

L'UI Son contient :

- prompt musical ;
- paroles ;
- mode instrumental ;
- durée ;
- qualité ;
- aide au prompt par genre, ambiance, énergie, instruments et voix ;
- structure musicale.

Structure musicale

Les structures doivent être écrites entre crochets pour éviter qu'elles deviennent des paroles chantées :

```
[Intro – Atmosphere]
[Verse 1 – Main Theme]
[Verse 2 – Variation]
[Bridge – Tension Build]
[Outro – Dissipation]
```

Conseil pour musique organique

Pour éviter un son trop synthétique, orienter le prompt vers :

```
organic orchestral instruments, real strings, acoustic piano, natural dynamics, warm
room, no harsh synthesizer
```

Les LoRA ou adapters audio peuvent aider si le backend les supporte, mais l'UI actuelle indique qu'ACE-Step applique plutôt un adapter à la fois.

15. Rôles personnalisés

POP Switcher permet à l'utilisateur d'ajouter des rôles custom.

Un rôle custom a :

- un identifiant stable ;
- un nom affiché ;
- une couleur ;
- une icône ;
- une fiche outil pour The One.

Fiche outil The One

La fiche outil contient :

```
Description
Idéal pour
Moins adapté à
```

Exemple pour un rôle `logo` :

```
Description : modèle spécialisé logo et identité visuelle premium.
Idéal pour : logos minimalistes, pictogrammes, marques, icônes simples.
Moins adapté à : photo réaliste générale, personnages complexes, scènes détaillées.
```

The One lit ces informations dans sa liste d'outils. Si une demande parle de logo, il peut alors choisir `logo` plutôt que `image`.

Pourquoi c'est important

Sans description, The One voit seulement un nom. Avec une fiche outil, The One peut comprendre :

- quel rôle est spécialisé ;
- quand l'utiliser ;
- quand ne pas l'utiliser ;
- quel rôle choisir si plusieurs modèles sont attribués.

16. The One

The One est l'orchestrateur long terme de POP Switcher.

Il sert à réaliser une tâche complexe en plusieurs étapes. Il ne doit pas être une UI Réflexion renommée : c'est un mode projet durable.

Principe général

Quand l'utilisateur écrit une mission, The One :

1. analyse la demande ;
2. découpe le projet en étapes ;
3. choisit les rôles utiles ;
4. crée un dossier de projet ;
5. exécute les étapes branchées automatiquement ;
6. laisse les étapes non branchées validables manuellement ;
7. garde les artefacts ;
8. produit un récapitulatif final.

Dossier mémoire The One

```
~/Applications/WorkFlow/Mémoire/The One/the_one_projects.json
~/Applications/WorkFlow/Mémoire/The One/<projectID>/
```

Chaque projet a son dossier propre.

Interface The One

L'UI The One affiche :

- mission ;
- discussion The One ;
- projet actif ;
- statut ;
- dossier projet ;

- étapes ;
- artefacts ;
- rôles utilisables ;
- projets récents ;
- mémoire ;
- journal ;
- orientation utilisateur.

Nouveau projet propre

Le bouton `Nouveau projet propre` prépare une mémoire visible propre sans supprimer les anciens projets.

Utilisation recommandée :

1. cliquer sur `Nouveau projet propre` ;
2. écrire une nouvelle mission ;
3. cliquer sur `Lancer projet`.

Reprendre un projet

Les projets récents sont cliquables. Quand un projet est repris, The One affiche un message résumant :

- le nom du projet ;
- l'objectif ;
- l'avancement ;
- le dossier ;
- ce que l'utilisateur peut faire ensuite.

Modification d'un projet existant

Si une nouvelle demande ressemble à une modification, The One injecte le contexte du projet actif :

- demande initiale ;
- objectif ;

• fichiers ;

• artefacts ;

• chemins.

Cela permet de dire :

```
Rends l'affiche plus premium et ajoute une variante fond clair.
```

sans repartir de zéro.

Rôles branchés automatiquement

À la version v2.28 :

• `chat` et `theOne` peuvent produire du texte via llama.cpp ;

• `reasoning` peut être utilisé ou retomber sur `theOne` / `chat` ;

• `image` peut lancer une génération image via le runtime existant ;

• `code`, `audio`, `imageEdit`, `vision` et rôles custom sont utilisés prudemment comme étapes texte si leur exécution spécialisée n'est pas encore branchée ;

• `video` est volontairement exclu de l'automatisation The One.

17. Host Bridge et switchCORE

Le Host Bridge permet de piloter POP Switcher par HTTP local sans cliquer dans l'interface.

Principes

• serveur local sur `127.0.0.1` ;

• port configurable ;

• token Bearer obligatoire ;

• JSON simple ;

• aucune dépendance au texte visible des boutons ;

• aucune dépendance à System Events pour changer de rôle ;

• compatible usage minimisé ou sans fenêtre au premier plan autant que possible.

Routes principales

```
GET  /host/ping
GET  /host/status
GET  /host/roles
GET  /host/endpoints
POST /host/runtime/stop
POST /host/runtime/open-ui
POST /host/role/switch
POST /host/role/switch-and-start
POST /host/image/generate
GET  /host/jobs/{id}
POST /host/tasks/dispatch
GET  /host/tasks
GET  /host/tasks/{id}
```

Exemple d'appel image

```
{
  "prompt": "French man on the beach",
  "width": 1024,
  "height": 1536,
  "steps": 8,
  "guidance": 1.0,
  "outputDirectory": "/Users/florentbrunet/Applications/WorkFlow/Image",
  "restoreRoleID": "chat"
}
```

Jobs

Les tâches longues utilisent des états de job :

```
queued
preparing
switching_role
runtime_starting
running
saving_artifact
restoring_chat
done
failed
```

Un job contient :

- jobID ;
- kind ;
- roleID ;
- status ;
- startedAt ;
- finishedAt ;
- progressMessage ;
- artifactPath ;
- errorMessage.

18. Mémoire et historique

POP Switcher utilise plusieurs mémoires séparées.

Mémoire conversationnelle

Les UI de conversation gardent l'historique visible des échanges.

Exemples :

- UI Chat : conversations ;
- UI Réflexion : discussion + contexte web ;
- UI The One : discussion projet ;
- UI Code : compte rendu et historique par projet.

Mémoire web

L'UI Réflexion conserve :

- derniers résultats DuckDuckGo ;
- pages ouvertes ;
- extraits ;
- résumé de mémoire ;
- état de tâche autonome.

Mémoire The One

The One conserve :

- projets ;
- étapes ;
- artefacts ;
- dossier ;
- journal ;
- orientations utilisateur.

Mémoire fichiers

Les fichiers produits restent dans WorkFlow :

- images dans `WorkFlow/Image` ;
- audio dans `WorkFlow/Audio` ;
- vidéos dans `WorkFlow/Vidéo` ;
- code dans `WorkFlow/Code` ;
- projets The One dans `WorkFlow/Mémoire/The One`.

Compression mémoire

Le principe recommandé est de ne pas garder indéfiniment tout le texte brut. Quand la mémoire devient longue, il faut la résumer :

- garder les faits utiles ;
- garder les chemins de fichiers ;
- garder les décisions ;
- supprimer les logs répétitifs ;
- conserver les artefacts sur disque.

Le but est de préserver la continuité sans faire exploser le contexte du modèle.

19. Architecture technique

Fichiers principaux

```
GLOB/MainWindowController.swift
GLOB/AppPaths.swift
GLOB/HostBridgeController.swift
GLOB/ImageRuntimeController.swift
GLOB/RuntimeController.swift
GLOB/SystemMonitor.swift
GLOB/LocalizedText.swift
GLOB/LogStore.swift
Scripts/build_universal.sh
```

MainWindowController.swift

C'est le fichier central historique. Il contient encore beaucoup de logique :

- interface principale ;
- rôles ;
- assignation modèles ;
- UI WebKit ;
- réglages ;
- The One ;
- dispatch de tâches ;
- génération image ;
- gestion code ;
- logique audio/vidéo ;
- synchronisation runtime.

Ce fichier a grossi au fil du projet. Pour un développeur, le meilleur futur chantier est de le découper progressivement en modules.

AppPaths.swift

Définit :

- nom public `POP Switcher` ;
- ancien nom `GLOB` ;

- dossier Application Support ;
- dossier WorkFlow ;
- chemins LLM, Audio, Image, Vidéo, Code, LoRA ;
- miroir des préférences.

C'est le point de référence pour éviter d'éparpiller les fichiers.

HostBridgeController.swift

Gère le serveur local HTTP :

- cycle de vie ;
- token Bearer ;
- routes ;
- statut ;
- jobs ;
- appels vers MainWindowController.

WebKit UIs

Plusieurs UI spécialisées sont rendues comme interfaces WebKit internes. Ce choix permet d'avoir des interfaces riches sans dépendre d'une app web externe.

Important :

- l'utilisateur interagit avec l'UI ;
- mais la logique doit rester pilotée par Swift ;
- les tâches ne doivent pas dépendre de clics externes ;
- les popups système sont évitées quand elles cassent le flux.

Build universel

Le script :

```
./Scripts/build_universal.sh
```

compile :

```
arm64
x86_64
```

puis fusionne avec `lipo` pour produire :

```
POP Switcher-v2.28.app
```

20. Historique et logique de développement

POP Switcher a été construit progressivement.

Phase Glob

Au départ, l'application s'appelait Glob. Elle servait surtout de tableau de bord local pour lancer des modèles et afficher l'état système.

Phase SwitchAI

Le projet a ensuite évolué vers SwitchAI. Cette phase a ajouté :

- les rôles rapides ;
- l'assignation modèle par rôle ;
- les UI spécialisées ;
- l'image ;
- la retouche ;
- la vidéo ;
- le son ;
- le Host Bridge local ;
- les jobs et artefacts.

Phase POP Switcher

Le projet a ensuite été renommé POP Switcher pour clarifier son rôle : un orchestrateur local capable de travailler seul ou avec switchCORE.

Cette phase a ajouté :

• le dossier WorkFlow ;

• le miroir des préférences ;

• une meilleure séparation fichiers utilisateur / dépendances techniques ;

• The One ;

• les fiches outil pour les rôles custom ;

• la logique de projets longs.

Conséquence technique

Le code est encore très concentré dans `MainWindowController.swift`. C'est normal pour un prototype devenu produit, mais ce n'est pas l'état idéal à long terme. Le développeur qui reprend le projet doit éviter de tout réécrire d'un coup. La bonne stratégie est de créer progressivement des contrôleurs spécialisés, puis de déplacer la logique par petits blocs vérifiés.

21. Dépendances principales

Homebrew

Gestionnaire de paquets macOS. Utile pour installer des outils comme Python, cmake, git, ffmpeg ou dépendances de build.

Python

Nécessaire à ComfyUI, Wan, LTX et certains workflows image/vidéo.

llama.cpp

Backend pour les modèles texte et multimodaux GGUF.

Utilisé par :

• Chat ;

• Code ;

• Réflexion ;

• The One ;

• Vision selon modèle.

stable-diffusion.cpp

Backend image local pour certains modèles GGUF image.

Utilisé par :

• Image ;

• Retouche Image ;

• SDXL Turbo ;

• Z-Image ;

• Qwen-Image ;

• FLUX.2.

ComfyUI / Wan video stack

Backend graphique/nodal pour workflows vidéo.

Utilisé par :

• Wan ;

• LTX ;

• certains workflows vidéo avancés.

ffmpeg

Outil vidéo/audio pour assembler, encoder ou convertir les résultats.

ACE-Step / acestep.cpp

Backend audio local utilisé pour générer musique et son.

22. Modèles recommandés dans l'aide système

```
Chat : Gemma 4 chat, llama.cpp
Vision : Gemma 4 vision, llama.cpp
Code : Qwen texte / code, llama.cpp
```

```
Image : Z-Image-Turbo GGUF, stable-diffusion.cpp + Z-Image VAE + Qwen3 4B encoder
Retouche : Qwen-Image-Edit 2511 Q3_K_M GGUF, stable-diffusion.cpp + Qwen-Image VAE +
Qwen2.5-VL 7B encoder
Audio : ACE-Step-1.5 GGUF, ACE-Step runtime + LM + encoder + VAE
Vidéo : Wan 2.2 TI2V-5B Turbo GGUF ou Wan Diffusers selon workflow
```

Autres modèles listés :

```
SDXL Turbo GGUF
FLUX.2-klein GGUF
Qwen-Image GGUF
Wan 2.2 TI2V-5B GGUF
Wan 2.2 T2V-A14B GGUF
LTX2.3-10Eros GGUF
```

23. Permissions macOS

POP Switcher essaie de limiter les permissions.

À éviter

Mettre les modèles dans :

```
~/Documents
~/Desktop
~/Downloads
```

peut provoquer des demandes d'autorisation macOS répétées.

Recommandé

Utiliser :

```
~/Applications/WorkFlow/LLM
```

ou un dossier externe choisi clairement dans les réglages.

Permissions qui peuvent rester nécessaires

Selon les usages futurs :

• Full Disk Access si une app externe doit lire des bases Messages ;

• Automation vers Messages ou Mail ;

• Screen Recording pour analyse d'écran ;

• accès fichiers si l'utilisateur choisit un dossier protégé.

POP Switcher ne doit pas utiliser Accessibility pour changer de rôle en interne.

24. Conseils performance

Choisir un modèle adapté à la RAM

Règle simple :

• un GGUF 4 Go est beaucoup plus réaliste qu'un GGUF 20 Go sur une machine limitée ;

• `Q4_K_M` est souvent un bon compromis ;

• `Q5_K_M` peut être plus qualitatif mais plus lourd ;

• les modèles vidéo peuvent saturer la machine même à basse résolution.

Génération light

Le mode génération light doit chercher à laisser des ressources au Finder et aux autres apps :

• réduire nombre de threads ;

• réduire résolution ;

• réduire steps ;

• éviter les grosses générations vidéo pendant le travail ;

• limiter les tâches concurrentes ;

• éviter les scans répétés de tailles de dépendances.

Vidéo

Pour tester :

```
640 x 360
4 secondes
16 FPS
```

```
4 steps si turbo
20 steps maximum pour un premier test non turbo
```

Si une génération dépasse plusieurs heures, arrêter et réduire le workflow.

25. Dépannage

Le modèle n'apparaît pas

Vérifier :

- le modèle est dans le dossier LLM scanné ;
- le fichier est bien `.gguf` ou le format attendu ;
- le rôle accepte ce type de modèle ;
- le scan a été relancé ;
- les préférences de dossier LLM ne pointent pas vers un ancien chemin.

Le runtime ne démarre pas

Vérifier :

- dépendance installée ;
- binaire détecté ;
- modèle assigné ;
- port libre ;
- logs Application Support.

Host Bridge inaccessible

Vérifier :

- bridge activé ;
- port correct ;
- token Bearer ;
- fichier status ;

• log `host_bridge.log` ;

• pas de conflit de port.

Image ou vidéo incohérente

Vérifier :

• modèle adapté au mode utilisé ;

• VAE et text encoder installés ;

• résolution raisonnable ;

• prompt clair ;

• réglages recommandés ;

• workflow validé.

L'UI ne charge pas le modèle

Vérifier :

• rôle assigné ;

• UI assignée ;

• runtime lancé ;

• contexte compatible ;

• chemin modèle non protégé ;

• état du rôle dans l'écran principal.

26. Recommandations développeur

Priorité 1 : modulariser

Le fichier `MainWindowController.swift` devrait être progressivement découpé :

```
RoleRegistry.swift
SettingsController.swift
TheOneController.swift
ReasoningController.swift
CodeWorkspaceController.swift
```

```
ImageTaskController.swift
AudioRuntimeController.swift
VideoWorkflowController.swift
```

Priorité 2 : stabiliser les contrats

Définir clairement les contrats :

- rôle ;
- runtime ;
- UI ;
- job ;
- artefact ;
- projet ;
- dépendance ;
- mémoire.

Priorité 3 : rendre les rôles custom plus puissants

La fiche outil custom est un premier pas. À terme, un rôle custom devrait pouvoir déclarer :

```
roleID
title
runtimeKind
inputSchema
taskKindsSupported
artifactKindsProduced
toolDescription
bestFor
limitations
defaultUI
defaultRestoreRoleID
```

Priorité 4 : renforcer The One

Étapes futures :

- brancher Code comme vraie exécution fichier ;

- brancher Retouche Image ;
- brancher Audio ;
- garder Vidéo désactivé tant que les workflows sont trop longs ;
- ajouter audit automatique d'artefact ;
- mieux gérer les reprises après redémarrage ;
- compresser automatiquement les mémoires de projet.

27. Ce qu'il faut retenir

POP Switcher est devenu un hôte local de rôles IA. Son utilité vient de trois couches :

1. UI manuelles simples pour l'utilisateur ;
2. runtimes locaux spécialisés par rôle ;
3. orchestration avancée via Host Bridge et The One.

Le plus important pour maintenir l'app est de garder les rôles stables, les chemins propres, les dépendances explicites, les jobs lisibles et The One isolé du reste pour éviter de casser les UI existantes.