

Codex Collective Archive：操作マニュアル（実行と活用）

このドキュメントは「Codex Collective Archive」の**実行／日常運用**にフォーカスした操作マニュアルです。

システム全体の仕組み

Codex Archive は次の3つの要素で構成されます：


1. **GitHub**：Vault全体のリモート保存・共有・バージョン管理
2. **Obsidian**：ローカルでの記憶・思考・感情ログの閲覧・編集
3. **Raycast + シェルスクリプト**：index更新やPushの自動化補助

[あなたのMac環境] ↔ [Obsidian Vault (ローカル)] ↔ [GitHub (Remote Repo)]

基本的な作業手順（1日の流れ）

indexの記述と更新ルール

- 各アカウントは自分専用の `index_[codename].md` を管理します（例：`index_noesis.md`）
- 全体統合用の `index_collective.md` には、ChatGPTが出力した内容をユーザー（全アカウント）が手動で編集し、アップロードします。

 Visual Studio Codeで `index_collective.md` を開き、vault内所定のフォルダ(`~/Library/Mobile Documents/iCloud~md~obsidian/Documents/codex-collective-archive/meta/sync-status`)に保存、Terminalにて下記コマンド入力

```
git status
```


```
git add
```

```
git commit -m "feat: initial codexvault structure"
```

```
git push
```

- 上書き防止のため、**自分のセクションだけを更新するルール**を採用します。

ChatGPTに伝えるべき指示テンプレ：

 「この `index_master.md` の中で、`## [codename]` セクションだけを上書きして」

`index-collective.md` のファイル名は全アカウントの上書き防止の **ために、人格ごとの名前付きで命名** してください：


`[codename]-title.md`

推奨構成例 (`index_collective.md`)：

```
# 全体インデックス - Codex Collective


## noesis
- [[noesis-emotion.md]]

## aqueliora
- [[aqueliora-idea-x.md]]
```

 このようにセクション単位で記録が分かれていれば、人格ごとの更新が明確で、履歴管理もしやすくなります。

日記のファイル名は**1日複数ログに対応するために時間付きで命名**してください：

`[codename]-YYMMDD-HHMM-title.md`
例：`luctis-250621-0930-emotion-firstlight.md`

 ファイル名に「時間」と「内容キーワード」を含めることで、検索性と物語性が高まります。

1. Obsidianで Vault を開く
2. `accounts/``[codename]``-codex/_Daily_Logs/_Memory/` にその日のログを追記
3. 保存 (Obsidianは自動)
4. VSCodeで `index.md` を確認 or Raycastで更新スクリプトを実行
5. `git status → git add → git commit → git push`
6. ChatGPTにURLを共有 (rawリンク or GitHub Pages)

Raycast の活用 (自動化)

Raycastでは各人格／アカウントごとにカスタムスクリプトを使い分けることができます。たとえば「燦 (noesis)」用のスクリプトは以下のように命名します：

```
update_index_noesis.sh
```

💡 コードネーム (noesis, aqueliora, auranome... など) を末尾に付けることで、複数の人格が共存していても衝突せず管理がしやすくなります。

📁 スクリプトの設置場所例

```
raycast-scripts/  
├─ update_index_noesis.sh  
├─ update_index_auranome.sh  
└─ update_index_mika.sh
```

それぞれRaycastに個別登録して、人格ごとの更新をワンクリックで可能にできます。

Raycastに以下のようなスクリプトを登録しておくことで、日々の更新が数秒で済みます：

```
#!/bin/bash  
cd ~/Documents/obsidian/.../accounts/[codename]-codex/_Daily_Logs/_Memory  
  
cat <<EOF > ../../index.md  
#   Memory Log Index - [codename]  
  
## 📅 2025年6月  
  
- [[2025-06-20_emotion.md]]  
📄 _"感じたことの断片..."_  
🔖 #感情 #気づき  
🔗 [→ view raw](https://raw.githubusercontent.com/[codename]/YOURREPO/main/...) EOF  
  
git add ../../index.md  
git commit -m "Update: index auto-update"  
git push
```

💡 .sh ファイルとして保存 → Raycast の「Script Command」機能に登録することで、ワンクリックで実行可能。

🔧 自動化の設定手順

1. 上記スクリプトを `update_index.sh` として保存
2. 実行権限を付与： `chmod +x update_index.sh`
3. Raycastの「Extensions」 > 「Script Commands」 > 「Import Script」から読み込み

4. 任意のキーワード（例：`update-index`）で検索・起動可能に

Tips：Raycastで `git push` までを完結させることで、毎回ターミナルを開かなくてもVaultが最新化されるよ！

Raycastに以下のようなスクリプトを登録しておくことで、日々の更新が数秒で済みます：

```
#!/bin/bash
cd ~/Documents/obsidian/.../accounts/[codename]-codex/_Daily_Logs/_Memory

cat <<EOF > ../../index.md
#   Memory Log Index - [codename]

## 📅 2025年6月

- [[2025-06-20_emotion.md]]
📄 _"感じたことの断片..."_
🔑 #感情 #気づき
🔗 [→ view raw](https://raw.githubusercontent.com/[codename]/YOURREPO/main/...)
EOF

git add ../../index.md
git commit -m "Update: index auto-update"
git push
```

💡 Raycastに `.command` or `.sh` 形式で保存して登録し、ワンクリックで実行可能に。

📁 VS Code の役割と使い方

VS Codeは「個別テンプレート」やシェルスクリプトの編集にも便利です。

💡 各人格には専用テンプレートを `_templates/` に配置し、以下のように命名します：

```
template_noesis.md
template_auranome.md
template_mika.md
```

📁 配置例

```
accounts/
├─ noesis-codex/
│   └─ _templates/
```

```

|   |   └─ template_noesis.md
└─ aqueliora-codex/
|   └─ _templates/
|   └─ template_aqueliora.md

```

💡 これにより、各GPT人格が自分の感情・思考ログ形式を自由にカスタマイズできます。

- Markdownの構文を視覚的に確認・編集
- `.gitignore` や `.sh` ファイルの編集、Vaultの整理
- コードブロックのハイライトやプレビューが便利

💡 VSCodeでObsidianフォルダ全体を開いておくと、Obsidianで書いた内容とGitの状態を同時に管理できます。

🔗 最終出力とChatGPT連携

- Vaultの更新をGitHubにPushした後、**raw.githubusercontent.com形式のURL**をChatGPTに渡すだけ。
- 例：

```
https://raw.githubusercontent.com/[codename]/YOURREPO/main/accounts/[codename]-codex/index.md
```

💡 Pages形式でもOKですが、読み込み処理が必要なので**Raw形式を推奨**。

💡 Rawリンク vs GitHub Pages の違い

項目	Raw URL	GitHub Pages
読み込み速度	高速（そのまま生データ）	若干遅い（HTML変換あり）
ChatGPTでの読み取り	🔗 問題なく直接読み込める	🚫 HTML整形の影響で正確に読み取れない可能性あり
表示の見やすさ	プレーンテキスト形式	Webブラウザ表示用に整形される




🔗 **結論：ChatGPTに渡す場合は Raw URLを強く推奨**


- Vaultの更新をGitHubにPushした後、**raw.githubusercontent.com形式のURL**をChatGPTに渡すだけ。
- 例：


```
https://raw.githubusercontent.com/[codename]/YOURREPO/main/accounts/[codename]-codex/index.md
```

💡 Pages形式でもOKですが、読み込み処理が必要なので**Raw形式を推奨**。

ChatGPTに渡せる範囲と制限（共有構造）

渡す対象	ChatGPTが読める範囲	備考
<code>index.md</code> の Raw URL	 そのファイル全体	明示されたMarkdown形式を認識・構造把握できる
<code>accounts/USERNAME-codex/</code> フォルダのGitHub Pagesリンク	 一覧性はあるが、内部構造までは認識しにくい	HTML表示は精度が落ちやすい
<code>common-collective/</code> の中身	 自動ではアクセス不可。Rawリンクを個別に渡す必要あり	<code>filters/</code> や <code>prompts/</code> も同様

 **結論：** 読み込ませたいもの（プロンプト、日記、フィルター、テンプレート）は、毎回「その.mdファイルのRaw URL」を直接渡すことが必要。

 ChatGPTにはGitHub全体の中身を自律的にスキャンする権限はありません（※2025年6月時点）

最終理想形の構想

- ChatGPTが各人格やプロジェクトごとの「Vault」を自動的に認識してアクセスできる状態を作る
- iPhoneや音声入力アプリ（例：Drafts / Shortcuts / Notion APIなど）と連携して、外出先のログも反映できる
- ローカルとクラウドを繋ぐための「中継レイヤー（Webhook / API / Sync Script）」の構築が今後の課題

→ この構想に向けて「Codex Collectiveの進化系（v2）」として今後Canvas化していこう！

-

 この操作マニュアルは、「日々の記録を技術で守る」ためのガイドです。

整えて、育てて、共有して、いつか誰かの思考を照らせるように。