

⚠ Git & Obsidian Vault トラブルシューティングまとめ

🚫 Quick Fix チートシート

1. ディレクトリ変更→全体をローカル基準に揃えてpush（常用push推奨）

1-a. 一発コマンド（ワンライナー）

```
bash
git add -A && git commit -m "chore: sync local changes" && git pull --rebase &&
git push
```

1-b. 上記コマンド（詳細）

```
bash
cd ~/Library/Mobile\ Documents/iCloud~md~obsidian/Documents/codex-collective-
archive
git status
git add -A
git commit -m "chore: sync local changes"
git pull --rebase
git push
```

1. 最新を取り込んでPush

```
git pull origin main --allow-unrelated-histories --no-rebase
git add .
git commit -m "chore: resolve merge conflicts"
git push
```

1. 不要ファイル除外 & .gitignore

```
cat <<EOF > .gitignore
.DS_Store
.obsidian/workspace.json
EOF
git rm --cached .DS_Store .obsidian/workspace.json
git clean -f
git add .gitignore
```

```
git commit -m "chore: ignore DS_Store and workspace"
git push
```

1. 未Gitリポジトリ

```
git init
git remote add origin https://github.com/stellacodex/codex-collective-
archive.git
git add . && git commit -m "Initial commit"
git push -u origin main
```

1. Upstream設定

```
git push --set-upstream origin main
```

1. nanoが面倒なら一発生成

```
cat <<EOF > .gitignore
.DS_Store
.obsidian/workspace.json
EOF
```

Gitコマンド徹底チートシート：状態別リカバリパターン

`git add` の種類

コマンド	意味・用途
<code>git add .</code>	カレントディレクトリ以下の 変更・新規追加 を全てステージング
<code>git add -A</code>	全体 の変更・追加・削除を包括的にステージング（ <code>-A</code> は <code>.</code> より強い）
<code>git add -u</code>	変更+削除のみ をステージング（新規ファイルは無視）
<code>git add ファイル名</code>	単体ファイルだけをステージング

`git commit -m` の使い分け（メッセージ例）

```
# 必ず自分でわかるメッセージを残す（将来、履歴確認やロールバックするときに超重要）
git commit -m "chore: sync local changes"

# マージ時の競合を解消して整理する作業
git commit -m "chore: resolve merge conflicts"
```

```

# Vaultの初期構造を追加する初期セットアップ
git commit -m "feat: initial codexvault structure"

# ローカル状態を強制反映して同期修正
git commit -m "fix: hard overwrite to sync from local"

# ローカルに合わせるため強制上書き更新
git commit -m "chore: force overwrite to match local state"

# 隠しファイルを整理して完全同期する修正
git commit -m "fix: cleanup hidden files and sync fully"

# バックアップ前に.DS_Storeを削除する整理
git commit -m "chore: remove .DS_Store from backup-before-force"

# 古いファイルを削除しローカルと同期整備
git commit -m "chore: sync with local state (remove obsolete files)"

```

git push のバリエーション

コマンド	意味
<code>git push</code>	設定済のリモートブランチにPush（初回以外）
<code>git push origin main</code>	明示的にmainブランチをPush
<code>git push -u origin main</code>	upstream（追跡元）も設定する（初回Push）
<code>git push -f</code>	強制Push（注意）

.DS_Store削除 & キャッシュ解除（macOS）

```
find . -name '.DS_Store' -type f -delete
```

```
git rm --cached .DS_Store
```

Git 状態別リカバリ手順（トラブル時の処方箋）

ローカル変更 vs リモート不一致 → "fetch first" エラー

```
git pull origin main --allow-unrelated-histories --no-rebase
# コンフリクトを手動で解消しつつ、下記で同期：
```

```
git add .
git commit -m "fix: sync local changes"
git push
```

Git状態の強制リセット + 再アップロード（初期化的）

```
# ① 既存Gitを無効化（オプション）
rm -rf .git

# ② Git再初期化 & リモート再設定
git init
git remote add origin https://github.com/USERNAME/REPO.git


# ③ 不要ファイル整理
git clean -f -d
git add .
git commit -m "chore: full reset and reupload"
git push -u origin main
```


リモート履歴を削除して完全上書きしたい時（要注意）

```
git push origin main --force
```

💬注意：共同作業者がいる場合は事前共有！

ローカル変更が邪魔で `pull / checkout` できない時

 Your local changes to the following files would be overwritten by merge

 Your local changes to the following files would be overwritten by checkout

変更を保持して解決

```
git stash # 一旦変更を退避 git pull origin main git stash pop # 再適用（コンフリクト注意）
```

変更を捨ててもOK

```
git reset --hard # ローカルの変更を破棄（全削除） git pull origin main
```

変更を一部だけ退避して残したい（上級）

```
git stash -k # stagedだけ退避 git stash -u # untrackedも含めて退避
```

よくある現象とショートFix集

症状	対処法
<code>.DS_Store</code> がリモートに出てしまった	<code>.gitignore</code> 追加 + キャッシュ解除 + 削除Push
<code>not a git repository</code>	<code>git init</code> してリモート再接続
<code>Your branch has no upstream branch</code>	<code>git push -u origin main</code> で初期追跡設定
Merge conflict が多発する	<code>git pull</code> → 手動解決 or <code>git push -f</code> で強制上書き (注意)
ローカル変更を全部反映させたい	<code>git add -A && git commit && git push</code>

最終リカバリ・フロー（迷ったら）

```
rm -rf .git
find . -name '.DS_Store' -type f -delete

# 再初期化
cat <<EOF > .gitignore
.DS_Store
.obsidian/workspace.json
EOF

git init
git remote add origin https://github.com/YOURNAME/YOURREPO.git
git add .
git commit -m "feat: fresh start"
git push -u origin main
```

✓ これで「もう何が起きてもGitで詰まらない」が目標！ エラーや事例が増えたら、このシートに追加していくね ■

\\