

# 非エンジニアのためのGitHub基本操作

「push」と「pull」をマスターして同期の仕組みを理解する

---

Copyright © SPENDA Corp. All Rights Reserved.

## GitHubにおける「情報の同期」の仕組みを理解する

「push」と「pull」の役割を明確にし、チーム開発で事故を防ぐための「デイリーワークフロー」を学ぶ

pushとpullをマスターして、チーム開発を安全に

**GitHubでは、情報は常に2つのリポジトリ（貯蔵庫）を行き来します**

リモートリポジトリ（共有場所）：GitHubのサーバー上にある「マスターデータ」で、チーム全員が見る場所。ローカルリポジトリ（自分の作業場）：自分のPC内にある「自分専用の領域」で、納得いくまで自由に編集できる場所。

2つのリポジトリで情報を管理する仕組み

# 「push（プッシュ）」：成果を共有する

Confidential

自分の手元で行った作業を、GitHub上の共有場所に「押し出す」操作です

pushの前に、何を変えたかのメモ（履歴）を残すコミット（Commit）が前提。自分の成果がチーム全員に見えるようになり、複数の修正を「ひとまとめ」にして安全に反映できるメリットがあります。

成果をチームに共有する「押し出す」操作

# 「pull（プル）」：最新を取り込む

Confidential

**GitHub上にある最新の変更を、自分の手元に「引っ張ってくる」操作です**

他人が更新した内容を自分のPCに反映させる情報の同期。古いデータで作業を始めると、後で矛盾が起きやすくなるため、「まずpullから始める」のが鉄則です。

最新状態を取り込む「引っ張ってくる」操作

# コンフリクト（競合）とは何か？

Confidential

同じファイルの同じ行を、2人が同時に編集した際に起こる  
「確認作業」です

エラーではなく「安全装置」。GitHubが勝手に判断せず、「人間が選んでください」と警告してくれます。こまめにpush/pullを行い、差分を小さく保つことや、「今からここを直します」というチーム内コミュニケーションで回避できます。

安全に編集するための「確認作業」

## 「先祖返り」や「上書きミス」を防ぐための4ステップ

【朝】 まずpull：チームの最新状態を手元に入れる。【日中】 作業＆コミット：自分のPCでこまめに履歴を残す。【完了前】 念のためpull：自分が作業中に誰かが更新していないか確認。  
【夕方】 最後にpush：自分の成果をチームに共有する。

pullで始まり、pushで終わる安全なリズム

**pushは「共有場所へのアップロード」、pullは「最新状態のダウンロード」**

pullで始まり、pushで終わるリズムを身につけることが重要です。Next Action：テスト用のリポジトリで、1行だけのテキストファイルを「push」して、ブラウザで編集後に「pull」するキヤッチボールを試してみましょう！

pullで始まり、pushで終わるリズムを身につける

# お問い合わせ

運用作業やサポートを依頼したい方は下記よりご依頼頂けます。

- **Web** : <https://spendacorp.com/contact.html>
- **Email** : [contact@spenda-c.com](mailto:contact@spenda-c.com)

無料相談（30分）、アカウント診断申込など、お気軽にご相談ください。