これを知らないと迷子になる

GIT用語サバイバル

DESIGN ENGINEERING

Git 概要の説明

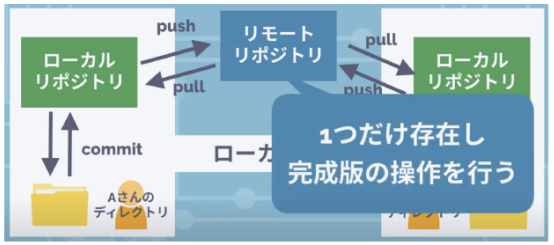
コードの変更履歴を記録・管理できるツール

ファイルのどこを、いつ、誰が、どう変えたかをすべて記録できるので、
過去に戻ったり、複数人で**安全に開発**したりできます。

📂 リポジトリ（Repository）

ソースコードを管理する場所

リポジトリとは、Gitで管理される**プロジェクトのデータの入れ物**です。
中には、ファイルだけでなく、**過去の変更履歴**も全て保存されています。
ローカルリポジトリ：自分のPCにある
リモートリポジトリ：GitHubなどにあるオンラインの保管庫

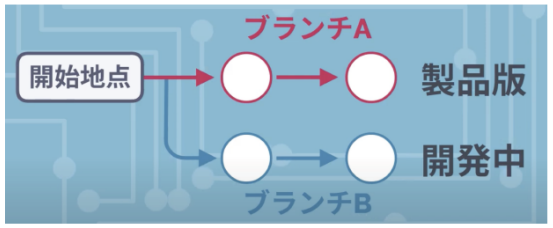



```
main　← 安定したコード
 └── feature/header ←header作業用ブランチ
 └── feature/footer ←footer作業用ブランチ
```

ブランチ（Branch）

開発を分岐させる仕組み

ブランチとは、作業用の分かれ道のこと。
機能開発やバグ修正を「main」から**分けて作業**することで、
本番コードに影響を与えず**安全に開発**できます。

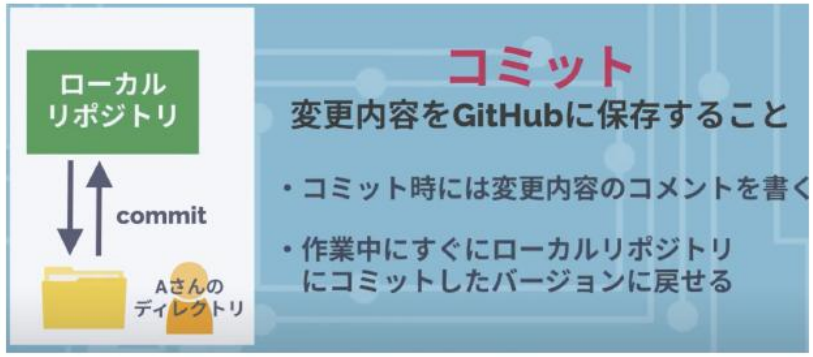


💾 コミット（Commit）

コードの変更履歴を記録すること

コミットとは、**変更した内容を記録として保存する**操作です。
ゲームでいう**「セーブポイント」**のイメージです。

```
git add .
git commit -m "ログイン画面のデザイン追加"
```

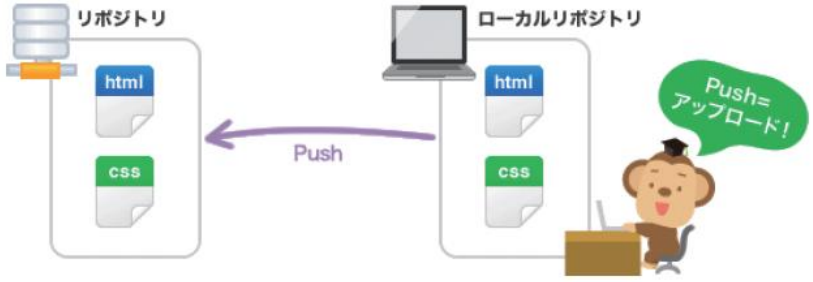


🚀 プッシュ（Push）

ローカルの変更をリモートに送信すること

自分のPCで**コミットした変更**を、GitHubなどの**リモートリポジトリ**にアップロードする操作です。

```
git push origin main
```

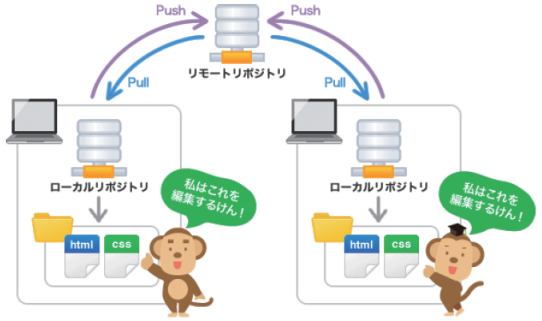


プル(Pull)

リモートの変更をローカルに取り込むこと

他の人がGitHub上で行った変更を、自分のPCに反映させる操作です。

```
git pull origin main
```

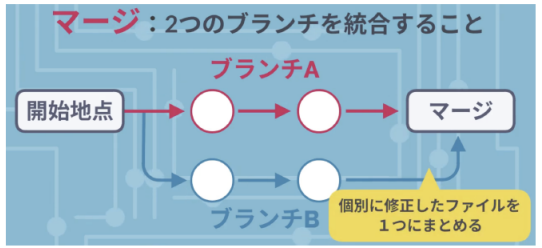


マージ(Merge)

複数のブランチを統合すること

機能ごとに分けたブランチを、最終的にmainブランチなどに統合する操作です。

```
git merge feature/login
```



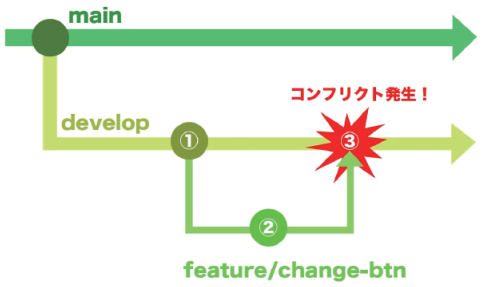
⚠ コンフリクト(Conflict)

マージ時の競合。手動で解消が必要

同じ場所を複数人が違う内容で編集していた場合、Gitはどれが正しいか判断できず「コンフリクト」が起きます。

例:

```
<<<<<<< HEAD
ここが自分の変更
=======
ここが他の人の変更
>>>>>>> feature/other
```



🚀 リリース（Release）

開発したシステムを公開すること

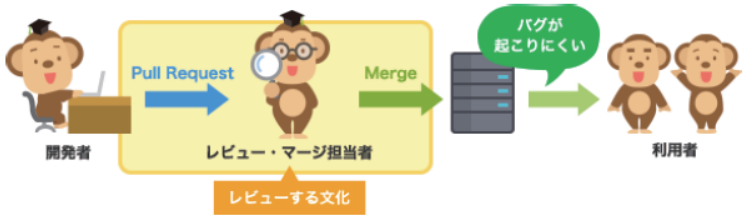
開発が終わったアプリや機能を、ユーザーに使ってもらえる状態にして公開することです。GitHubでは、特定のコミットにバージョンタグ（例：v1.0.0）をつけて、リリースノートも書けます。

📩 PR（Pull Request）

他人にレビューしてもらい、コードを統合する仕組み

PRは「Pull Request（プルリクエスト）」の略で、GitHub上で行うマージの申請手続きです。

チーム開発で「コードレビュー」や「承認」を行うための基本的な流れです。



→

1. 作業ブランチを作る
2. コードを書いてコミット・プッシュ
3. GitHubでPRを作成
4. 他の人がレビューしてOKならマージ！

LINE公式アカウントにて
最新情報配信中！

変化を楽しみ、自分らしく未来へ。

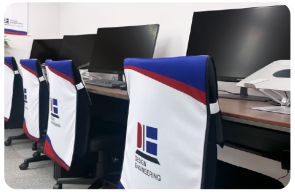
デザインエンジニアリングは、
挑戦するエンジニアの一歩を応援する会社です。

“好き”や“ワクワク”をそのままキャリアに変え、
自分の可能性を信じて前へ進む人には、無限のチャンスが広がっています。

失敗も学びに変え、仲間と共に笑い、共に成長しながら、
毎日が少しずつ楽しくなる未来へ。
未経験でも大丈夫。あなたの最初の一歩を、心からお待ちしています！

DESIGN ENGINEERING

イベント・セミナー開催中！



無料で自由に使える
学習＆カフェスペース開放中！

デザインエンジニアリング 検索

URL：https://design-engineering.jp/

カジュアル面談・エントリーは
こちらから！

LINE ID：@749gaovb