

Git / GitHub 入門講座

RICORA Programming Team

今日やること

#RICORA

- バージョンコントロールシステムであるGitの使い方を学ぶ
- GitリポジトリのホスティングサービスであるGitHubを使ってみる

Gitってなに？

“ Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. ”

- 要約すると「Gitは早くて効率的なオープンソース分散型バージョンコントロールシステム」
- それって実際どういうこと？ 🤔

Gitが作られた背景

#RICORA

- 最近人間味が出てきたことLinuxの製作者Linus Torvaldsが作った
- メールで送られてくるパッチを早くマージしてかつトラッキングするツールがほしかったらしい



Linus Torvalds ([©Gigazine.net](https://www.gigazine.net/))

メリット

- 早い
 - サーバーから拾ってくるときにそんなに時間がかからない
- ログを追える
 - このファイルはどこで変更されて...みたいなことがわかる
- 複数人での開発に向く
 - 各人が開発した成果を持ち寄りやすい
- 管理するファイルを選ぶ
 - `.DS_Store` とかを含まないようにできる

Gitを用いた開発フロー

0. `clone` とか `fork` & ブランチ分ける
1. `add` & `commit` 実質的な開発
2. `push`
リモートサーバーに現在のリポジトリの状態を通知
3. Pull Request (GitHub利用)
main / masterに機能追加してください!と頼む
4. `merge` 機能追加
5. 0-4の繰り返し

GitHubってなに？

- Gitリポジトリのホスティングサービスの1つ
 - Git ≠ GitHub
- リポジトリ置き場としての機能だけではなく、チーム開発に便利な機能がたくさん
 - 便利なのでリモートリポジトリにはGitHubが使われることが多い
- ↓はGitHubのキャラクターのOctocat



Clone & Fork

- `clone` はその名の通りクローン=コピーをとってくる
- めっちゃ `clone` することが多い 基本構文は

```
$ git clone (repo url)
```

- `fork` はもとのリポジトリを自分のところにコピー
 - そこから自分で機能追加とかできる

Clone & Fork

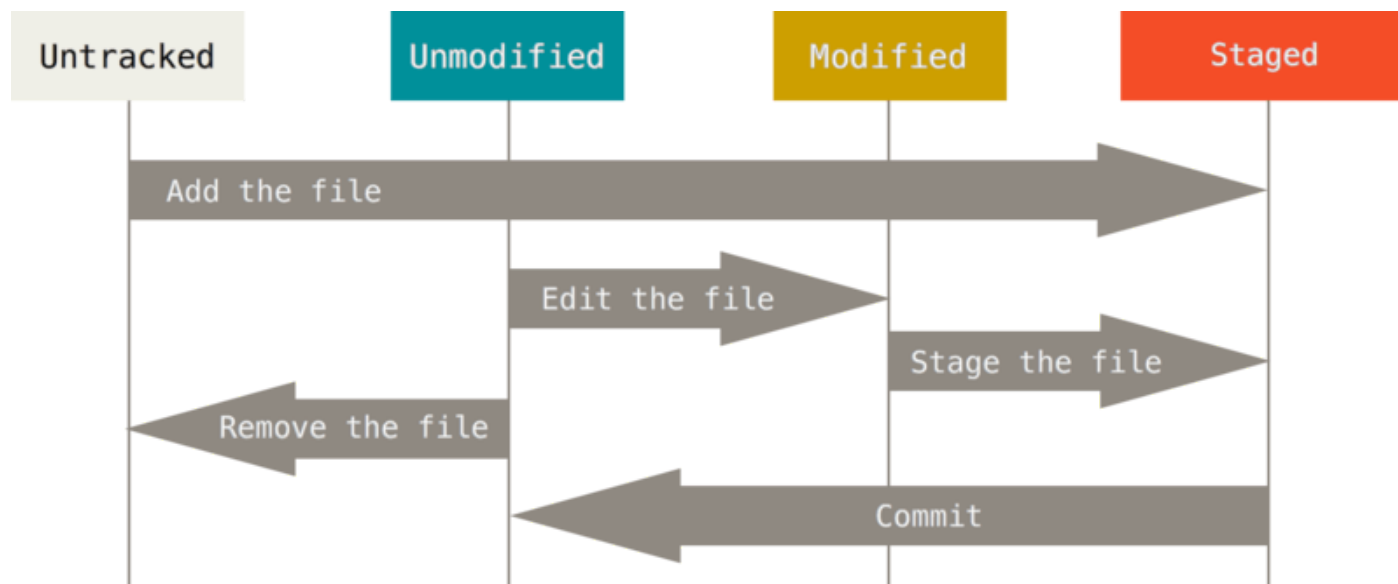
- これで開発の準備ができる
- 例えば `log` をしてみる（あとでこれが何をするか説明）

```
$ git log -p
```

- なんかログっぽいのが出る

Add & Commit (大事！)

- **add** : Gitに管理を頼む 変更を加えたファイルを選ぶ
- **commit** : 現在のスナップショットを保存
このスナップショットについてメッセージも書く(ログ)



([@git-scm.com](https://git-scm.com))

Add

- どのファイルを `git` に管理してもらうかを指定できる
- 絶対に無視したいファイルもある → `.gitignore` で指定

1. ファイル指定で `add` (推奨)

```
$ git add (something)
```

2. ファイル全体を `add` (非推奨)

```
$ git add .
```

Commit

- 今の状態を保存するように頼む
- コミットログを書く=なにを変更したのかを書く
- コマンドは

```
git commit -m (message)
```

- 過去のコミットはこれで追う

```
git log
```

Branch

- これだけだとDriveとかとあんまり変わらない 特色は*Branch*

Gitのブランチ

- ブランチ?
 - 別の世界線 = 別の履歴のこと 、(*°д°)ノカイバー
 - ブランチごとに `add` / `commit` する
 - 履歴を分離させることで機能別に開発ができる
- 機能ごとにBranchを分割 => master / mainに追加を繰り返す
 - 例: `docs/readme` とか

Branch

- 切り替え

```
git switch (branch name)
```

- 新規作成

```
git switch -c (branch name)
```

- あとは `man git-branch` で検索

んにゃぴ...よくわかんないっす

#RICORA

- それはそう
- Git
 - 様々なブランチにコミット → mainとかに取り込んでもらう
コミットの履歴がある
- Driveとか
 - 適当にファイルを入れる → 何が変更されたのかわからない
規模がでかくなると削除とかでぐちゃぐちゃになりがち

➡ Git使ったほうがよさそう（と思ってほしい）

リモートサーバーとかの話

- もちろん当時からLinuxは複数人で開発されていた
- Gitの威力はリモートサーバーがあるからこそ発揮される

Clone

#RICORA

- 再掲。リモートサーバーからリポジトリごと拾ってくる。

```
git clone
```

Push

- サーバーに変更分を通知
- あっち側にもBranchがあるので指定が必要（最初だけ）

1. 現在のBranchからPushしたことがない

```
git push --set-upstream (remote branch name)
```

2. Pushしたことがある

```
git push
```

Fetch / Pull

ここらへんは名前だけ一旦覚えておく

- `fetch`
 - リモートサーバーのBranchのHEADを拾ってくる。
- `pull`
 - Fetch & Merge する。とりあえず更新するときに使うみたいなイメージ

Pull Request

さっきちょろっと出てきたあれ。既存のものに何か追加するときを使う

Pull Requestの出し方

0. 先にブランチを **push** しておく
1. githubのリポジトリの右上の **Pull Request** をクリック
2. 左上のcompareから自分のブランチを選ぶ。baseはmainにする
3. 変更したことの太字を書いて、 **Submit Pull Request** をクリック
4. githubのリポジトリに戻って、 **Pull Request** の欄に自分が書いたものがあるか確認

全体の総括

#RICORA

0. `clone` & ブランチ分ける `switch`
1. `add` & `commit` 実質的な開発
2. `push`
リモートサーバーに現在のリポジトリの状態を通知
3. Pull Request (GitHub利用)
main / masterに機能追加してください!と頼む
4. `merge` 機能追加
5. 0-4の繰り返し

余談: Pull Requestの出し方

#RICORA

- 実は出し方が2種類ある

- ブランチを使う方法

別のブランチに変更を加える

用例: 複数人での開発

- **fork** する方法

既存のリポジトリを複製して自分で変更を加える

用例: OSSとかの機能改善

- 今回は上の方です・サークルでなんかやるときはブランチで

やってみよう

実際に触ってみよう 😊

やること

1. 環境構築して `git` を使えるようにする
2. 演習とかで `git` に触れてみる

Gitの前に

- パッケージマネージャーを導入する
- git, python, hugoなどのCLIツールをパッケージマネージャー経由でインストールする
- WindowsはChocolatey
 - <https://alg-slides.tus-ricora.com/pm.html#6>
- MacOSはHomebrew
 - <https://alg-slides.tus-ricora.com/pm.html#8>

GitHubの準備


- GitHubアカウントの作成
 - ない人はここから
 - <https://github.com/join>
- RICORA Organizationへの招待
 - 招待するのでGitHubのアカウント名を教えてください
 - 部員は[ここから](#)新入部員を**Member**で招待

GitHubの準備

- GitHubにSSH公開鍵を登録しよう
 1. ホームディレクトリで `ssh-keygen -t ed25519` を実行
 2. なにも入力せずEnter
 3. 公開鍵 `~/.ssh/id_ed25519.pub` の内容をコピー
 4. <https://github.com/settings/ssh> に公開鍵を登録
- 詳細はalg-wikiに書いたのでみてね
 - <https://alg-wiki.tus-ricora.com/ssh-tutorial/>

やってみよう

#RICORA

- 演習を作りました
 - <https://github.com/RICORA/git-tutorial-2022.git>
 - README.mdにやること書いてある
 - たぶん破壊できないけど破壊しないでね
- わかんなかったら聞いて 
- スライドを開いて参照しよう


Gitクライアントについて

#RICORA

- SourcetreeやGitKrakenなど、Git操作をGUIで行えるアプリケーションもあります
- GUIでGit操作が行えるので、Gitのイメージが掴みにくい初心者でも、直感的な操作が可能になり、とっつきやすい...かも？
- **Gitが使えるようになるのが目的なので、どうしてもコマンド操作が苦手という人は調べてみてください**

告知

#RICORA

- Git練習用のリモートリポジトリを用意しました
 - <https://github.com/RICORA/sandbox>
 - こっちは何をやっても 
 - でも履歴全消しとかはやめてね
 - 後日練習したいときに使ってね
- 座学だけだと意味不明だと思うので、とりあえず使いながら覚えよう

ご清聴ありがとうございました

RICORA