#RICORA

Git / GitHub 入門講座

RICORA Programming Team

はじめに

- バージョンコントロールシステムであるGitの使い方を学ぼう
- Markdownの使い方と二本立てで大変だと思いますが...

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

- 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)

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べる
  - `.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

#RICORA

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

ここでGitが使えるか確認

#RICORA

- あとで演習があります。
- `git` コマンドを実行して、オプション一覧が出てくることを確認。
- だめそうなら近くの人に助けてもらってね

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

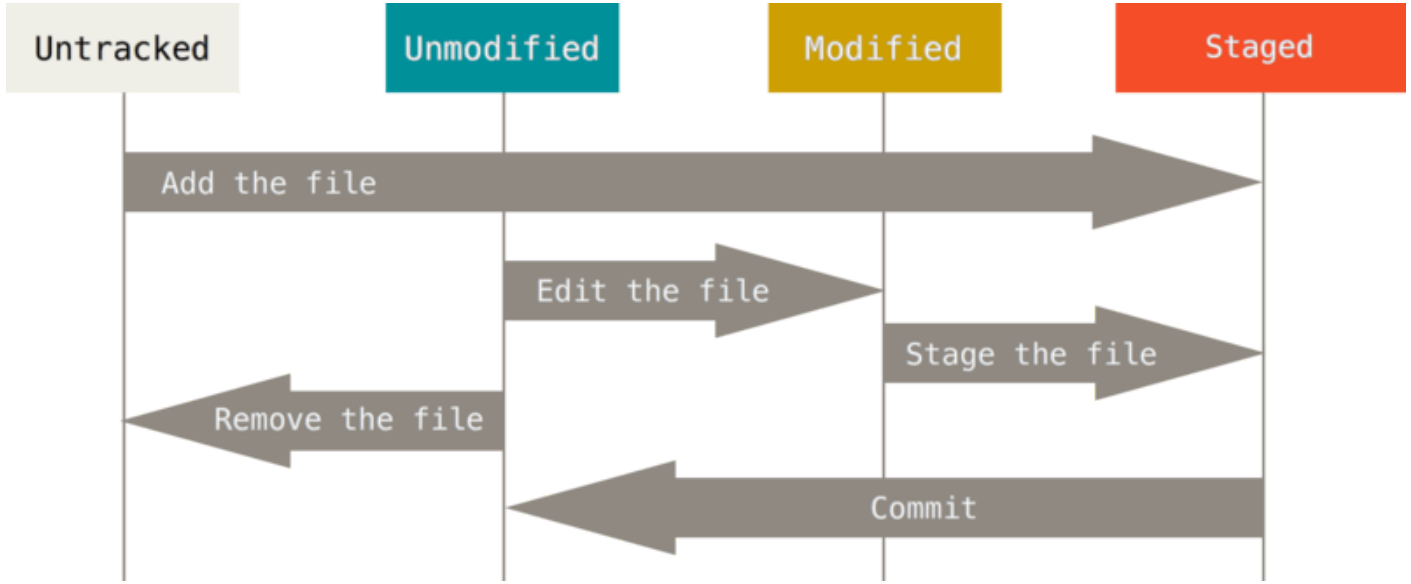
- これで開発の準備ができる
- 例えば `log` を見てみる（あとでこれが何をするか説明）

```
$ 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)

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 ヽ(*ﾟдﾟ)ノｶｲﾊﾞｰ
  - 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

#RICORA

- 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

- 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

全体の総括

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

やってみよう

- 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
  - 何をやっても🆗
    - でも履歴全消しとかはやめてね
- 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
- わからないことがあったら近くの人にきいてね

ご清聴ありがとうございました

#RICORA