GitHubの使い方について

プルリクエストを投げてマージする、
プロジェクト管理の幾つかの方法。

GitHubアカウントの準備

- https://github.com/RyoYoshida/html
  - こちらのリポジトリを利用する。
  - 会社メールのアカウントがない人は作成する。

GitHubDesctopを利用する。

- https://desktop.github.com/
  - GitHub公式のGit管理ツール。
  - 「余計な機能を持たずシンプル」な特徴。
  - GitHubと連携して様々な機能が利用できる。
- まずはこれをダウンロード・インストールする。
- 先ほどのリポジトリをクローンする。

まずは適当にコミットしてみます。

適当にプルリクエストを投げます。

- 各自一斉に好きなようにHTMLに機能を追加しましょう。
- コンフリクトが発生する。

正しいプロジェクト管理方法を理解する

- https://dev.classmethod.jp/articles/introduce-git-flow/
- https://qiita.com/yuki0410_/items/7c7fa20710dfd72b7d7a

実践

- developブランチを作成します。
- 様々なマージリクエストを投げます。
- 順番にコードをマージしていきます。

コンフリクト解決方法

- リベース方法
  - 基本的には自分に関わるプロジェクトはこちらを採用している。
  - コンフリクト解決の責務がリクエスト作成者に一任できる。
  - コミットログが非常に綺麗になる。
  - 必ずコマンドラインでの実行が必要。
    - **利用者全てに一定の技術レベルが必要。**
- マージ方法
  - レビューが終わった最後に元の最新ブランチをマージします。
  - コミットが汚れます。
  - コンフリクト解決の責務が管理権限者に依存します。
    - **管理者に業務が集中し開発進行が依存する。**

フォークを利用した3点リポジトリ構成

- 別途資料を参照。
- 管理者が担当するリポジトリ領域を絶対に侵食しない。
- 各自が自分のリポジトリ環境を自由に利用・破壊できる。
- プルリクエストの度にFetch-Rebaseが必要。
  - 現在の現場では自分がこれを採用した。

実践

- 各自リポジトリをフォークする。
- ここまでのブランチプッシュを試してみる。
- 差分のある状態でfetch-rebaseを試してみる。

最後に

- 開発環境の規模と技術習熟度によって適切な方法を選ぶ。
  - デザイナー等にはコマンドラインは一切利用できない。
  - GitFlowでは容易にdevelopの上書きや他人のfeatureブランチの浸食が起こる。
  - 3点リポジトリ構成で事故が起きたことはない。