github

開発開始時

変更を取ってくる

git fetch --all または　git fetch origin　←リモートの変更を取得

fetch後に、自分の変更を追加する  
git merge origin 【ブランチ名称】　←他者の変更とは別に自分の変更を追加し合流させる

git rebase origin 【ブランチ名称】　←他者の変更後に自分の変更を追加

ブランチを切る(すでに作成済みの場合は、移動する)

ブランチの名称は feature/XXXとなる。

git branch feature/【ブランチ名】　feature/XXXという名称のブランチを作成する

ブランチの移動

git checkout feature/【ブランチ名】

・コーディングやテストを行いファイルを作成・変更する

開発途中～終了時

追加の前にfetchとmergeで更新されていないか確認する。

追加・変更したファイルをリポジトリのファイルに入れる

git add 【追加ファイル名】で認識させる(すべての追加ファイルの場合は -Aを入れる)

branchがfeature/【ブランチ名】なことを確認する。(水色で表示されている)

ローカルリポジトリにコミットする。

git commit -m “[コミット種別] #プルリクエスト番号 要約”

例：git commit -m “[add] #1 〇〇機能の実装” (commitごとに何をやったかわかりやすくするため、毎回記述する)

●コミット種別

fix：バグ修正

add：新規（ファイル）機能追加

update：機能修正（バグではない）

remove：削除（ファイル）

リモートリポジトリへpush(絶対にmainにpushしない！)

git push origin feature/【ブランチ名】

開発完了後

Rull Requestを出す(完成した場合のみ)

タイトルを [ブランチ名]　要約

内容(Write)には、概要とmerge受け入れ要件を書く

Create pull requestを押す

共通の機能の作業を行う場合は、ブランチ名を同じにして同じブランチでpush、fetchしあうのが望ましい(その場合はコンフリクトに気を付けること、ユーザフォルダ直下のYT41-EquipmentSystemのフォルダで作業している場合を除く)

ブランチを分けて行う場合は、以下の方法でデータを取得する

他人のブランチのデータを取得する場合

git clone -b ブランチ名

git@github.com:YT41-EquipmentSystem/YT41-EquipmentSystem.git