GitHub のすすめ

計数工学科数理情報工学コース B4・さくさくラズパイ @radi_bow

2016/12/31
更新 2017/4/14

1　GitHub とはなんぞや

GitHub[*1]とは，ソフトウェアを開発する際に役立つコード管理システムのことです．この Web 上のサービスを用いることによって，ソースコードの共有やバージョン管理を行うことが可能です．この稿では GitHub の使い方について，ざっくりと紹介をします．もっとも，Web 上にも http://www.backlog.jp/git-guide/ などわかりやすい紹介がたくさんあるので，そちらをじっくり読んだ方が使えこなせるようになるかとは思います．

2　使い方・一般的な話編

GitHub を使うためには，アカウントをまず作成する必要がありますが，それについては省略します．

アカウントを作りあなたのページに飛んだら，リポジトリを作成してみましょう．**リポジトリ**とは，ファイルを保存する場所のことと思って差し支えありません．リポジトリには 2 種類あって，ネットワーク上にありメンバーで共有する**リモートリポジトリ**と，あなたの PC 内にありあなただけが編集する**ローカルリポジトリ**があります．基本的には，ローカルで作業した内容を随時リモートに反映させる（これを**プッシュ**という）ことで，プロジェクトを進めていきます．Web 上で新しく作ったリポジトリは，当然のことながらリモートリポジトリです．この新しいリモートリポジトリ，ないし他の人が作った既存のリポジトリをあなたの PC に「ダウンロード」してローカルリポジトリを作成すれば，準備完了です．このリモートからローカルを作成することを，**クローン**と呼びます．

さて，ローカル上で何らかの実装を済ませたとしましょう．これを皆で共有するリモートリポジトリに反映させる（プッシュする）には，**コミット**と呼ばれる作業をしてあなたがファイルに対して行った変更内容をまとめる必要があります．無事にコミットができたら，いよいよリモートリポジトリへのプッシュです．この際，あなたがリモートからクローンしてから修正点をプッシュするまでの間に，運悪く他のメンバーもコードの同じ部分を編集しコミット・プッシュしていたと仮定しましょう．このままあなたのコミットが他のメンバーのコミットを上書きしてしまうと，他のメンバーの生み出した進捗はすべて台無しになってしまいます．そこでこのような場合には，GitHub はコードが「衝突」していることを教えてくれます．その指摘を見て，あなたはその衝突部分を適切な形に整える必要があるのです．このようにリモートとローカルの整合性を保ち

[*1] https://github.com/

ながらローカルの内容をリモートに反映させることを，**マージ**と呼びます．衝突が起きていない限り，マージは自動的に GitHub が行ってくれます．

もっとも，このようなコードの衝突が起こらないようにするために，こまめに他のメンバーがリモートに対して行ったプッシュの内容を，あなたのローカルにも反映させることは重要です．このリモートの内容をローカルに引っ張ってくることを，**プル**と呼びます．

GitHub での作業は，（プルして）コミットしてプッシュ（して，必要な場合手動でマージ）するのが 1 サイクルです．使い始めは慣れないでしょうが，健闘を祈ります．ちなみに GitHub は直観的な GUI でコミットやプッシュができるデスクトップアプリ[*2]も用意しているので，これを使うと幸せになれるかもしれません．

3　使い方・細かい話編

作業の流れは前節で述べた通りなのですが，画像処理班で作業するにあたり数点補足をします．その前に，**ブランチ**と呼ばれる概念について説明を加えなければなりません．

ブランチとは，作業内容を分岐させて平行に記録するためのものです．各メンバーは，master という大きな木の幹から自分のブランチを切って，そのブランチ上で作業するようにします．ブランチ上での作業やコミット・プッシュが済んだら，自分のブランチを master にくっつけ（これも**マージ**という），必要に応じて衝突を解消して master に自分の作業を反映させます．このブランチですが，今回は何のために切ったブランチかをはっきりさせるために，**ブランチ名をそのブランチでやっている作業内容にしてください**．これが 1 つ目のルールです．

またそれに関連して，今回は master **へ直接プッシュすることを禁止**します．じゃあ肝心の master に自分の変更を反映させるためにはどうすればいいんだということになりますが，そのときは変更点を他のメンバーにチェックするように要望する役割のある**プルリクエスト**というものを送ってください．他の人がプルリクを確認次第，その作業内容を確認して master にマージしてくれることでしょう．

最後に，各メンバーは**こまめなコミット**をするようにお願いします．一度にたくさんのコードを書いてコミットすると，そのコードが何のために書かれたものなのか分かりにくくなってしまいます．ある一つの機能を実装したら，ファイルを保存するのと同じような感覚でコミットすることをお勧めします．その際のコミットメッセージには，一言でいいので実装した内容を書いておいてください．

[*2] https://desktop.github.com/