プログラミング (http://techacademy.jp/magazine/topics/programming)

今さら聞けない！GitHubの使い方【超初心者向け】(http://techacademy.jp/magazine/6235)

開発者によってなくてはならないサービスとして**GitHub**があります。エンジニアにとっては当たり前のサービスですが、これからプログラミングの勉強を始める初心者にとってはよくわからないかもしれません。

そこで今回は、そんな初心者でも今日から使えるように解説していきます。ぜひ自分で操作しながら使い方を覚えてください。

本記事は、オンラインブートキャンプ (https://techacademy.jp/briefing-lp?utm_source=tamagazine&utm_medium=referral&utm_content=6235&utm_campaign=post) のGitHubのカリキュラムをもとに執筆しています。

目次

GitHubとは

GitHubとは、ソフトウェア開発プロジェクトのためのソースコード管理サービスです。

公開されているソースコードの閲覧や簡単なバグ管理機能、SNSの機能を備えており、開発者にとって無くてはならないサービスです。

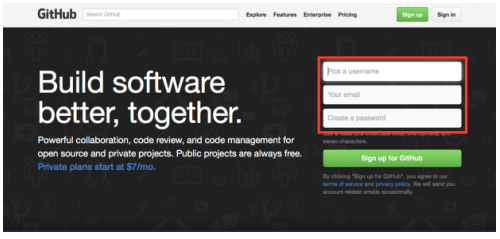
また、GitHubを使ってバージョン管理を行っている企業も多数あります。

GitHubのアカウント登録

それでは早速GitHubに登録してみましょう。

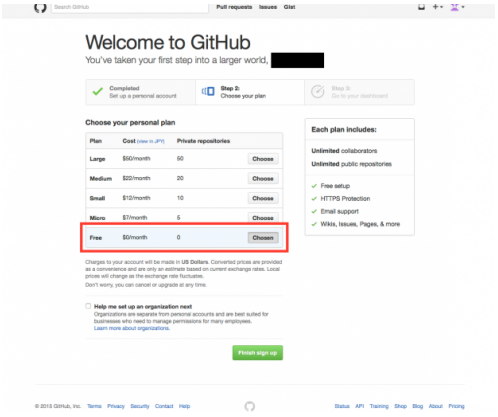
まずは、GitHubのトップページ (https://github.com/)にアクセスします。

ここで、ユーザ名とメールアドレス、パスワードを入力して、アカウント登録を行ってください。



続いて、プランを選択します。有料のプランもありますが、無料で使えるFreeプランがあります。

今回はFreeプランで登録するので、「Free」を選んでから「Finish sign up」ボタンをクリックします。



登録したメールアドレスに認証のメールが届きます。メールの内容に従いユーザ認証を行ってください。

以上でGitHubのアカウント登録は完了です。

GitHubを使う上で知っておきたい事前知識

早速GitHubを使っていきたいところですが、使い始める前に知っておきたい事前知識を3つ紹介します。

ここで紹介する言葉がまったくわからない場合は、理解しておきましょう。

事前知識1：ローカルリポジトリとリモートリポジトリ

リポジトリとは、ファイルやディレクトリの状態を保存する場所です。変更履歴を管理したいディレクトリなどをリポジトリの管理下に置くことで、そのディレクトリ内のファイルなどの変更履歴を記録することができます。

リポジトリは自分のマシン内にある「ローカルリポジトリ」とサーバなどネットワーク上にある「リモートリポジトリ」の2箇所にあります。基本的にローカルリポジトリで作業を行い、その作業内容をリモートポジトリへプッシュする流れで行います。

事前知識2：コミットとプッシュとは

最低限この2つは知っておきましょう。

- **コミット(commit)**：ファイルの追加や変更の履歴をリポジトリに保存すること
- **プッシュ(push)**：ファイルの追加や変更の履歴をリモートリポジトリにアップロードするための操作

事前知識3：ブランチ（branch）とは

ソフトウェアの開発では、現在リリースしてるバージョンのメンテナンスをしながら新たな機能追加やバグ修正を行うことがあります。このような、並行して行われる複数のバージョン管理を行うために、Gitにはブランチ（branch）という機能があります。ブランチは履歴の流れを分岐して記録していくものです。分岐したブランチは他のブランチの影響を受けないため、同じリポジトリ内でそれぞれの開発を行っていくことができます。

GitHubの使い方

それでは使い方を覚えていきましょう。

基本的なGitの作業は下記のような流れとなるので、GitHubを使って順番に紹介します。

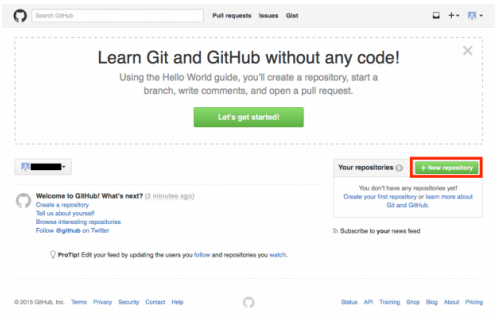
なお、1の作成は初回のみ行い、2から5を繰り返します。基本的に小さい作業の単位でコミットを行い、ある程度作業がひと段落した時にプッシュをするのが一般的です。コミットの作業がわかりやすいように、コミットメッセージを残しておくと、ログを追っていく時に役立ちますので覚えておきましょう。

1. リポジトリを作成（git init）、または複製（git clone）する
2. ファイルの作成、編集を行う
3. ファイルの作成/変更/削除をgitのインデックスに追加する（git add）
4. 変更結果をローカルリポジトリにコミットする（git commit）
5. ローカルリポジトリをプッシュしてリモートリポジトリへ反映させる（git push）

GitHubにリポジトリを作成する

まずはリポジトリを作成します。

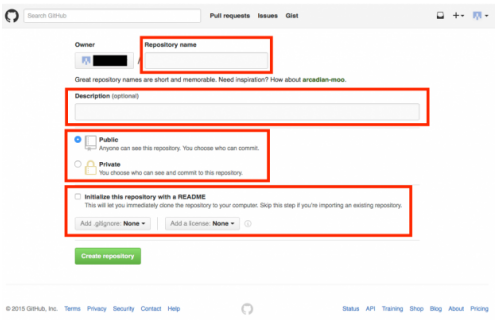
GitHubにログインした状態で、「New Repository」ボタンを押下します。



次に表示される画面では、「Repository name」の入力のあと、必要に応じて「Description」も入力します。

また、リポジトリの種類を「Public」か「Private」を選択します。この「Private」リポジトリは、有料会員のみ作成することが可能です。

最後に、リポジトリの中にあらかじめREADMEファイルを作成しておく場合は「Initialize this repository with a README」にチェックを入れます。.gitignoreやlicenseについては後で追加や変更ができますので、Noneを選択します。



必要項目の入力が終わり「Create repository」ボタンをクリックするとリポジトリの作成は完了です。

次の画面で、リモートリポジトリのアドレスが表示されますので、控えておいてください。

ファイルの作成、編集を行う

今回は、「hello.html」というファイルをローカルのPC上に作成した想定で進めます。

テキストエディタなどでHTMLファイルを作ります。

はじめに、ローカルのPC上にローカルリポジトリを作成します。

今回は「awesome」というディレクトリを作成することにします。

```
mkdir awesome
cd awesome
git init
```

このgit initコマンドはGitリポジトリを新たに作成するコマンドです。バージョン管理を行っていない既存のプロジェクトをGitリポジトリに変換する場合や、空の新規リポジトリを作成して初期化する場合に使用します。git initコマンドを実行するとカレントディレクトリをGitリポジトリに変換します。

ファイルの作成/変更/削除をgitのインデックスに追加する（git add）

先ほど作成した「hello.html」のファイルをローカルリポジトリに追加しましょう。

以下のコマンドでインデックスに追加します。
インデックスとは、リポジトリにコミットする準備をするために変更内容を一時的に保存する場所のことです。

```
git add hello.html
```

変更結果をローカルリポジトリにコミットする（git commit）

次に、インデックスに追加されたファイルをコミットします。

次に、インデックスに追加されたファイルをコミットします。
コミットとは、ファイルやディレクトリの追加や変更をリポジトリに記録する操作のことです。

```
git commit -m “add new file”
```

これで、リポジトリに対してファイルの追加が記録されました。

ファイルが追加されているか確認します。

```
git status
```

さらに、リモートリポジトリに反映させる前に、リモートリポジトリの情報を追加します。この情報は、先ほどGitHub上に表示された、リモートリポジトリのアドレスです。今回は例を示します。

```
git remote add origin https://github.com//awesome.git
```

ローカルリポジトリをプッシュしてリモートリポジトリへ反映させる（git push）

ローカルリポジトリの変更を、GitHub上にあるリモートリポジトリに反映させるため、以下のコマンドを実行します。

```
git push origin master
```

GitHubのユーザ名とパスワードを尋ねられますので入力してください。
これで、GitHubへプッシュしてリモートリポジトリへ反映させることができました。

よく使うGitのコマンド12

最後によく使うGitのコマンドを12紹介します。
既に紹介しているものもありますが、まずはここから覚えておきましょう。

git status

リポジトリの状態を確認するために使うコマンドです。
現在のブランチの名称や、追加・変更されたファイルやディレクトリの一覧を表示します。

```
git status
```

git add

ファイルやディレクトリをインデックスに追加するために使うコマンドです。
追加するときの[file_pattern]には、ファイル名やディレクトリ名を直接している他に、「*.txt」のように、ワイルドカードで複数対象を指定することもできます。

```
git add [file_pattern]
```

git commit

インデックスに追加されたファイルやフォルダの変更をリポジトリに書き込むために使うコマンドです。オプションを指定しないでこのコマンドを実行すると、コミットメッセージを記述するためのエディタが起動します。

エディタの使い方はそれぞれ異なりますので、簡単にメッセージを指定するには -m オプションを付けた後にダブルクォーテーションで囲ったメッセージを指定します。また、 -a オプションを指定すると、変更されたファイルを検出してインデックスに追加する動作も同時に行います。

```
git commit -am “A first commit”
```

git branch

ブランチに対して各種操作を行うために使うコマンドです。下記のように使います。

- git branch [branch-name]：ブランチの作成
- git branch：ブランチの一覧表示
- git branch –d [branch-name]：指定したブランチを削除

git checkout

ローカルリポジトリのブランチを切り替えるときに使うコマンドです。

```
git checkout [branch-name]
```

git log

ローカルリポジトリのコミット履歴を閲覧するために使うコマンドです。
-n オプションで履歴の表示数を指定できます。

```
git log –n 10
```

git grep

リポジトリのファイルの内容から検索したいときに使うコマンドです。

特定の語句が含まれているファイルを検索し、そのファイルのどこに語句が含まれているかを調べることができます。

```
git grep "検索単語"
```

git clone

既存のリモートリポジトリをローカルに落とすために使うコマンドです。
例えば、GitHubに公開されているリポジトリを自分のコンピュータへ落とすときに使います。

```
git clone [url]
```

git remote

リモートリポジトリを操作するために使うコマンドで下記のように使います。

- git remote：リモートリポジトリの名称一覧を表示
- git remote -v：リモートリポジトリの詳細一覧を表示
- git remote add [name] [url]：リモートリポジトリを追加
- git remote rm [name]：リモートリポジトリを削除

git reset

ローカルリポジトリのコミットを取り消すために使うコマンドです。
間違えてコミットしたり、修正漏れがあったときによく使います。

```
git reset –soft HEAD^
```

git merge

現在のブランチに対して、他のブランチで行った変更を取り込むために使うコマンドです。
以下の例では、ブランチbug-fixをmasterブランチにマージします。

```
git checkout master
git merge bug-fix
```

git pull

リモートブランチの変更を取り込むために使うコマンドです。
以下の例では、ローカルリポジトリのmasterブランチにリモートリポジトリoriginのmasterブランチを取り込みます。

```
git checkout master
git pull origin master
```

今回の記事は以上です。
ぜひご自身で使ってみてください！

Gitのインストール方法の記事 (http://techacademy.jp/magazine/5304)もあるので、合わせてご覧ください。

MENU

当メディアについて (http://techacademy.jp/magazine/about)
お問い合わせ (http://techacademy.jp/magazine/contact)

SEARCH

カテゴリーから探す (http://techacademy.jp/magazine/category)
アーカイブから探す (http://techacademy.jp/magazine/archive)
キーワードから探す (http://techacademy.jp/magazine/keyword)

SERVICE

オンラインブートキャンプ (https://techacademy.jp/)
プログラミング研修 (http://techacademy.jp/corporate/)
ワークショップ (https://workshop.techacademy.jp/)

MAIL MAGAZINE

新着記事などのメルマガを受け取る

hello@techacademy.jp 登録

(https://www.facebook.com/techacademy)
(https://twitter.com/techacademy)
(https://plus.google.com/+TechacademyJp)
(http://techacademy.jp/magazine/feed/)