‘14

Git の使い方

紅狼

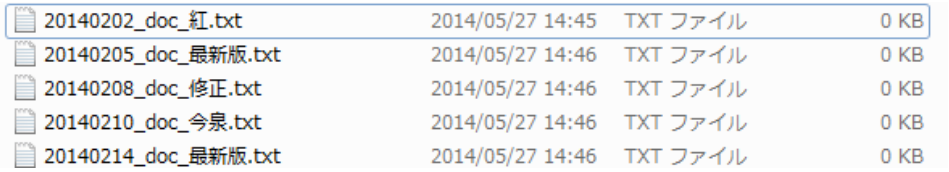
1. Git とは？

Git は，一言で言うと分散型バージョン管理システムといいます．離散型バージョン管理システムってなに？ Git を使うとどう便利なの？どうやってるの？ということをこの項では説明していきます．

1.1. Git の利便性

～*Git* がない世界～

プログラムのコードを書いていくと，“このプログラム，前は動いてたのにいろいろ追加してる間に動かなくなった！”ということが多々あります．(あって欲しくないのですが) では，節目節目で動くようなファイルを残しておけば，前の動く時のファイルに戻してそこから編集をすることができます．ではファイルを残してみましょう．



こうなりました．非常にカオスです ファイル名を間違えると何が最新版なのかよくわかりません．どのような編集をしたのかもよくわかりません．また，共同で開発をしていると，“自分が編集した部分を間違って他の人が上書きしてしまって気づかぬうちに自分のファイルが変わっていた．“ということもありえます．非常に恐ろしいです．

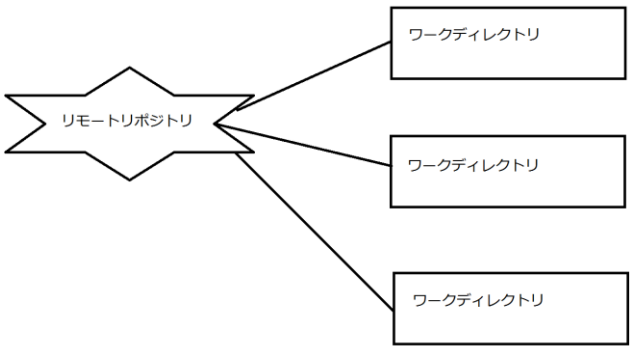
～分散型バージョン管理システム *Git* がある世界～

- Git では，他人が自分のファイルを編集するときに警告がでます．
- Git では，誰が，どのような編集をしたかをオフラインで(分散型)確認することができます．
- Git では，簡単に最新バージョンを取得，いつ編集したかもわかります(バージョン管理システム)

1.2. 状態を保存するリポジトリ

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．(どんな変更がされたかなどはここに入っています．) Git のリポジトリには，以下の 2 つがあります

- リモートリポジトリ
  サーバーを使って複数人で共有する
- ローカルリポジトリ
  ユーザーが使用するために個人の PC においておくリポジトリ(操作するのはこちらです)



リポジトリを２つに分けることによって，中途半端な状態でサーバーにアップロードせず，ちゃんと動作する状態を常にサーバーに残すことができます．こうすることで，プログラムをサーバーから引き出しても常に動く最新版のプログラムを取得することができます．

1.3. 変更を記録するコミット

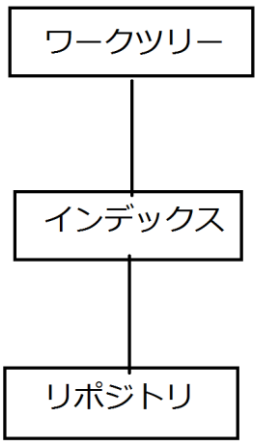
ファイルやディレクトリの追加・変更をリポジトリに記録することをコミットといいます．

コミットを実行すると，前回コミットした時の差分を記録したリビジョン(コミット)と呼ばれるものが作られます．また，コミットをするさいにメモ書きを残すことができます(変更内容の要約，理由など)

1.4. ワークツリーとインデックス

Git で，実際にユーザーが操作する部分をワークツリーと言います．リポジトリとワークツリーの間に，コミットするために準備をするインデックスと呼ばれる部分があります．このインデックスがあることにより，ディレクトリの中の一部のファイルのみをアップロードしたり，メモ書きなど，必要のないファイルのコミットを防ぐことができます．

注）インデックスに登録されていないファイルはコミットできません



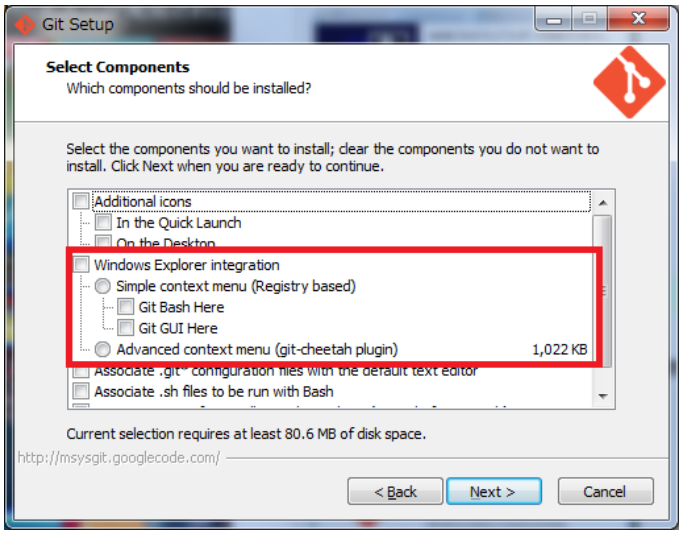
2. Git のインストール(Windows)，初期設定

2.1. Git のダウンロード(Next の回数等は ver.により変わる可能性あり(2014/03/20 現在:1.9.1))

2.1.1. http://git-scm.com/ の Download for Windows からダウンロードする.

2.1.2. ダウンロードした.exe ファイルを実行し，Next を押していきます下記の画像の部分に注意してください.

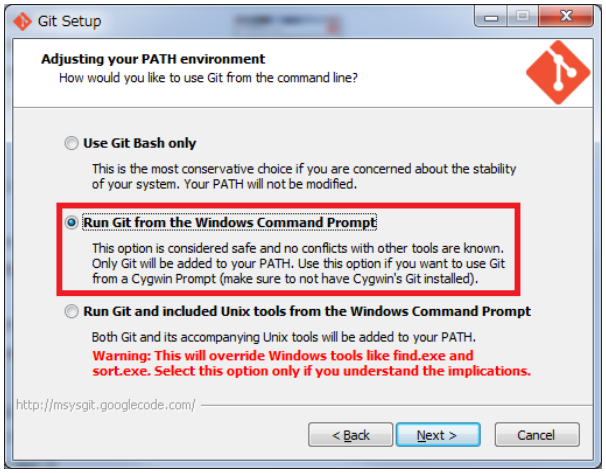
2.1.3. 下図では，好きな様にチェックを付けてもらって構いません．赤い四角のところをつけておくと便利かもしれません(この本(?)では扱いません)



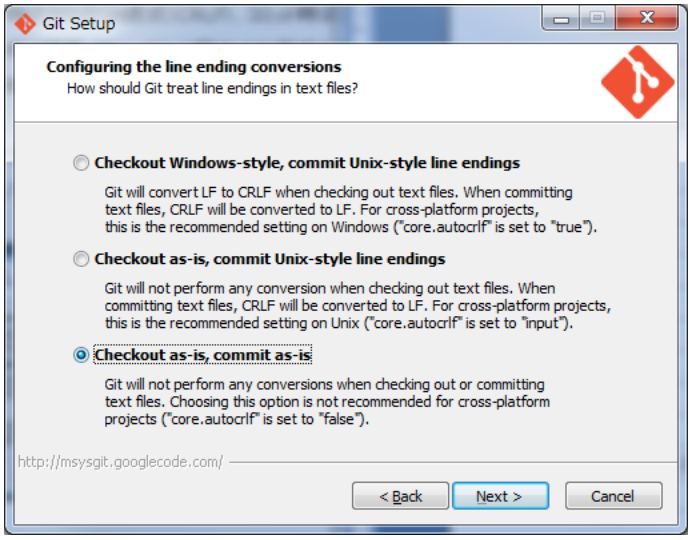
2.1.4. 下図では，赤い四角のところにチェックを付けるのがオススメです．上から簡単に説明すると

- ✧ GitBash というツールのみで Git を使う
- ✧ Windows のコマンドプロンプトでも使えるようにする
- ✧ Windows のコマンドプロンプトに Unix のコマンドを使えるようにする

となっています.



2.1.5. 最後に，下図でも好きなところのチェックを入れてもらって構いません．よくわからない場合は一番上がオススメです(ちなみに自分は下図のようになってました)



2.2. Git の初期設定

2.2.1. インストールしててできたフォルダの中に GitBash を実行します．

2.2.2. デフォルトのユーザー名とメールアドレス名を登録します．自分の場合

ユーザー名が kurenaif

メールアドレスが fujiwara_koh@zeus.eonet.ne.jp

なので

```
$ git config --global user.name kurenaif
$ git config --global user.email fujiwara_koh@zeus.eonet.ne.jp
```

となります．(わかりづらいですがハイフンは 2 個です)

```
$git config --global --list
```

でちゃんと入力できたかを確認するといいです．

3. [初級]Git の使い方

この項では”アップロード側”ではなく，“ダウンロード側”の説明をします

3.1. clone(git clone [URL] [ディレクトリ])

clone コマンドは既存のリポジトリをコピーしてきます．では，ネット上にあるリポジトリを clone してみましょう．(これには GitHub を使用していますが，今は特に気にしなくていいです．)

```
$ git clone https://github.com/kurenaif/HowToGit
```

ls コマンドでディレクトリが増えたか見てみましょう．(Unix コマンドつかえます)

```
$ ls
```

それがコピーしてきたディレクトリです．

3.2. pull(git pull)

pull コマンドでは，.git/config に設定されたリモートリポジトリの変更を取得(し，現在のブランチにマージします． ブランチ，マージは後で説明します)例を挙げるのが難しいので，複数人で push(後述)したり pull したりしてみて試してみましょう．

```
$ git pull
```

clone はリポジトリを持ってきて生成するのに対し，push は更新する．この違いは抑えておきましょう

4. Git の使い方

こちらでは Git の作り方，書く人が見る項目です．

4.1. リモートリポジトリの作り方

```
$ mkdir -p [リポジトリ名].git  (ディレクトリを作ります)
$ cd [リポジトリ名].git (ディレクトリの移動をします)
$ git --bare init  (git 用に初期化します)
```

いろいろあると思いますが，とりあえず次へ進んでみてください．

4.2. ローカルリポジトリの作成→リモートリポジトリの登録

```
$ mkdir [リポジトリ名] (.git は不要です)
$ cd [リポジトリ名]
$ git init
$ echo “[文字列]” > readme
$ git add readme (add:インデックスにファイルを登録します(1.4 参照) )
$ git commit –m “[メモ書きの内容]” (commit:コミット(1.3 参照)します)
$ git remote add origin [URL] (URL に origin というニックネームを付ける)
$ git push origin master (origin(上の URL)に master ブランチをアップロードする)
```

一気に書いてしまってすみません．最後の master ブランチと言うのは気にしないでください．後で説明します．よく分からなかったらとりあえず master ブランチでやっときゃあ大丈夫です．コマンドの説明は上記で十分じゃないかと思います．一つ言いたいことがあるので書いておきます．

> こまめに commit 慎重に push (commit はゲームで言うクイックセーブみたいなものです．残せば残すほどその地点に復帰しやすくはなりますが，過剰な commit は探すのが大変になるのでほどほどに・・・． push はみんなが見るリモートリポジトリにアップロードします．みんながダウンロードすることを頭において push しましょう．もちろんバグがあるものや，編集途中のものを push したらみんな困ります．あと commit したのを戻すのは簡単ですが，push したのを戻すのはめんどうです・・・(経験談))

ファイルをコミットしたいときは add した後，commit すればいいです．
必要があれば push すればリモートディレクトリに反映されます． （後でやるのでここでは説明は省略します 上のやつとほぼ一緒ですし．

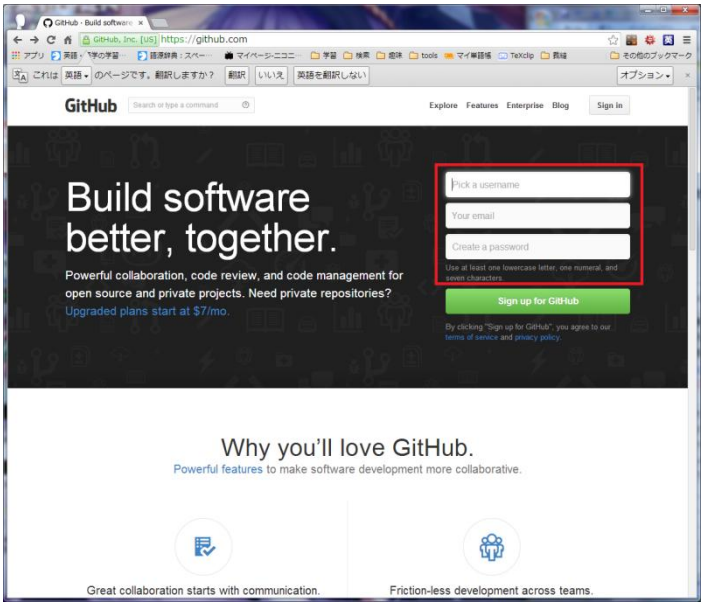
5. GitHub を使って練習してみよう

やってみないと意味がわかりません！とりあえずやってみましょう．研究室のサーバーは研究室でしかアクセスできない可能性があるので GitHub を借りましょう！（便利ですし）

5.1. アカウント登録

1) https://github.com/ にアクセスする

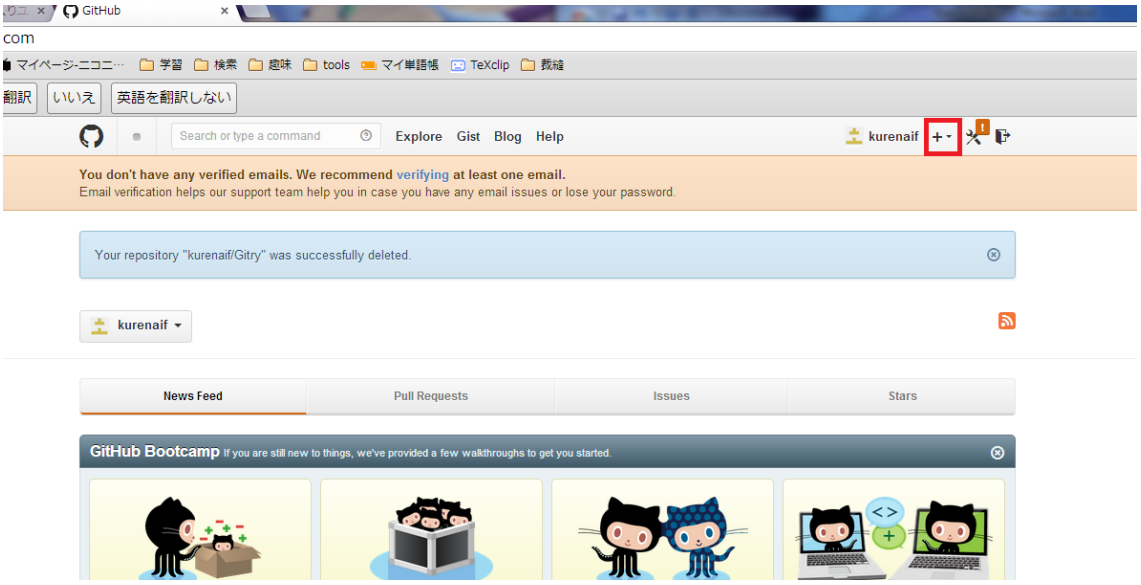
2) あとはここに適当に入力して Sign up for GitHub をクリックすれば OK です



5.2. リポジトリをつくろう

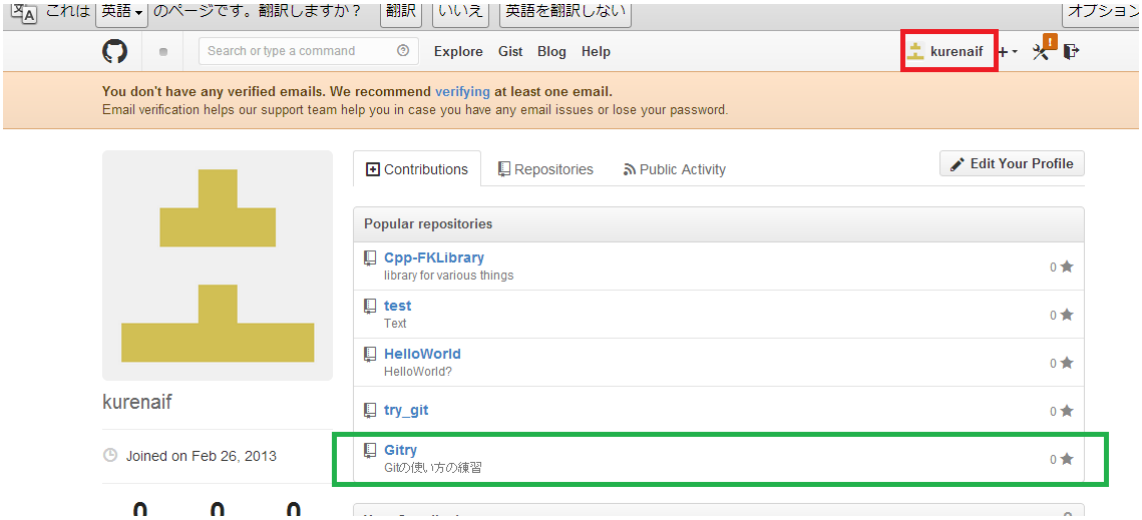
1) ログインした後,画面上部の赤い四角をクリックして New Repository をクリックしてください

2) 適当に入力した後, Create repository をクリックしてください



5.3. クローンしよう (自分は Gitry リポジトリを作りました.)

1) 赤い四角の部分をクリックした後，下の緑の四角の部分をクリックします. (次ページへ)



2) そのページの URL をコピーし, GitBash を起動します.

3) ワークディレクトリを作りたい場所に cd で移動し，clone します自分の場合，デスクトップに作って後で消したいので

```
$ cd Desktop
$ git clone https://github.com/kurenaif/Gitry
```

としました. (ちなみに GitBash 上でペーストは Shift+Insert です)

5.4. commit しよう

1) 出てきたフォルダの中に適当にファイルを作ってください. (適当に中も書いといてください)
(自分の場合, Hello.txt に World という内容にしました)

2) GitBash にて, cd コマンドでさっきできたディレクトリに移動してください

```
$ cd Gitry
```

3) いま，あなたのワークツリーには Hello.txt がありますが，インデックス上にはなにもありません(意味がわからない時は 1.4 の図を見てください.) commit するためには，インデックスに登録しないといけないので, add します.(1.4 参照)

```
$ git add --all
```

git add --all とすれば，追加したファイルや，変更したファイルをすべてインデックスに追加

できます. “4.2 ”でやったように，ファイル名を直接指定してやってもらっても構いません.

4) 続けて commit します.

```
$ git commit -m “First commit”
```

First Commit はメモ書きの内容です．好きなように変更してください.

5.5. push しよう

1) 最後に，push します．これでリモートリポジトリに反映されます．(origin というのは，GitHub のさっきの URL のことです. master は気にしなくていいです.)

```
$ git push origin master
```

2) 先ほどのページに戻ってみると，追加したファイル(Hello.txt が追加されていることがわかります.)

さらにファイルの変更があればまた add してから commit したらいいです．そして必要があれば push

6. より便利にするために

最低限上記の事ができればアップロードダウンロードはできますが，この項目では Git をより便利にするために書く項目ですちょくちょく更新すると思います.

6.1. Git の出力に色をつける

```
$ git config --global color.ui true
```

ユーザー設定です．便利です.

6.2. diff(差分を表示する)

冒頭で説明したどのような変化があったか．のコマンドになります．いくつか種類があるので紹介します.

```
$ git diff
```

index と working tree の差分を表示します. (working tree は，コミットする前のファイル, index は commit した後のファイル)

6.3. log(commit の log を表示する)

```
$ git log
```

6.4. branch

私が, Git で一番便利で，難しく，危険でもあると思う branch です．使うかどうかよくわからないのでとりあえず保留．これの内容はかなり重いので，これだけで 1 つ本(?)をつくろうと思います.

[参考資料]

サルでも分かる Git 入門	http://www.backlog.jp/git-guide
Wikipedia	http://ja.wikipedia.org/wiki/Git
git 公式ページ	http://git-scm.com/
Git の使い方(見出し)	http://transitive.info/article/git/
git リポジトリの作成方法	http://taichino.com/memo/1587