

# Git チュートリアル

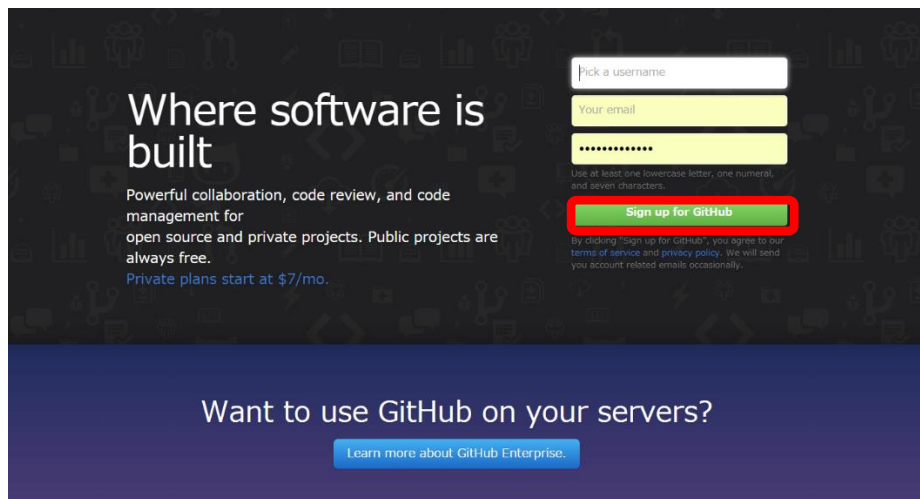
2015 年 10 月 14 日作成  
文責：井料研修士一回藤原

## 1. Github アカウントの作成

ここでは分散型バージョンシステム git を扱うためのサービス Github について説明します。

Step1 Github の公式サイトにアクセスします

<https://github.com/>にアクセスし、ページ上の「Sign up for Github」をクリック



Step2 アカウント作成

画面の[Username]、[password]に好きなユーザー名,パスワードを設定します。  
次に[Create an account]をクリック

Join GitHub  
The best way to design, build, and ship software.

☐ Step 1: Set up a personal account   ☐ Step 2: Choose your plan   ☐ Step 3: Go to your dashboard

Create your personal account

There were problems creating your account.

Username

Login can't be blank

You will occasionally receive account related emails. We promise not to share your email with anyone.

Password

Use at least one lowercase letter, one numeral, and seven characters.

By clicking "Create an account" below, you are agreeing to the Terms of Service and the Privacy Policy.

Create an account

You'll love GitHub

Unlimited collaborators  
Unlimited public repositories

☐ Great communication  
☐ Friction-less development  
☐ Open source community

その次に有償プランへのアップグレードについての説明が表示されます。ここでは無視してください。

### Step3 アカウント作成通知メールの確認

次に Github から確認のメールが届くのでメールの指示に従ってください。

#### Welcome to GitHub

You've taken your first step into a larger world, @ryu2200.

We're pretty certain this is the start of something really special, so we wanted to take a moment to personally welcome you to GitHub. We like to think of GitHub as the **best way to build and ship software**, and it's great to have you on board!

To make your life easier, we thought it might be helpful to point you to some awesome ways to use GitHub.

[GitHub Flow in the Browser](#)

Want to learn how to contribute to GitHub without learning Git? Learn how to branch, edit, and submit your first pull request all in the browser.

[Explore what is Trending on GitHub](#)

From open government data to web frameworks, our trending page is a great way to discover different ways people are using GitHub.

[Help & Support](#)

Feel stuck? It happens to the best of us. Head on over to <https://help.github.com> or [get advice and guidance from real humans](#). We're here to help.

You will also have a verification email in your inbox - please follow the link inside it so we can confirm your email address is really yours.

Thanks so much for your time — we're looking forward to seeing what you build with GitHub!



---

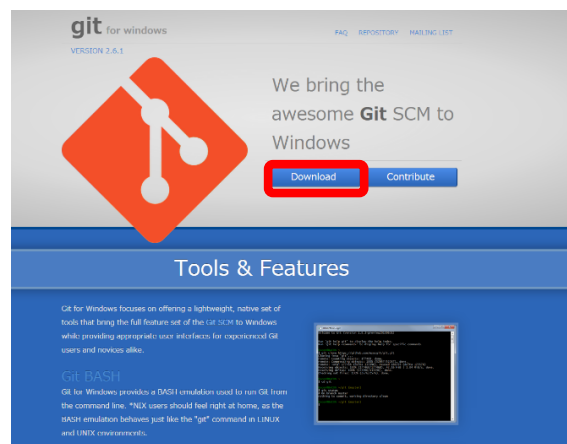
## 2. msysGit のインストール

Github アカウントを入手しただけではまだ `git` は使えません。Github はあくまでも分散型バージョンシステム `git` に対応した `web` サーバーを提供するサービスでしか無いからです。

ここでは `msysGit` という `git` があらかじめ組み込まれたシェル `mingw` などを含むパッケージを用いた `git` 環境の構築の手順について説明します。

Step1 `git` 公式サイトにアクセス

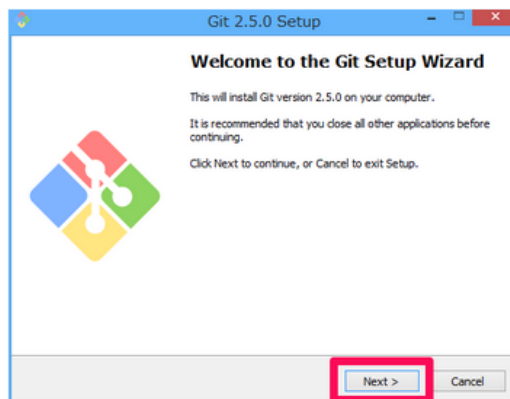
<https://git-for-windows.github.io/> にアクセスし「Download」をクリックします



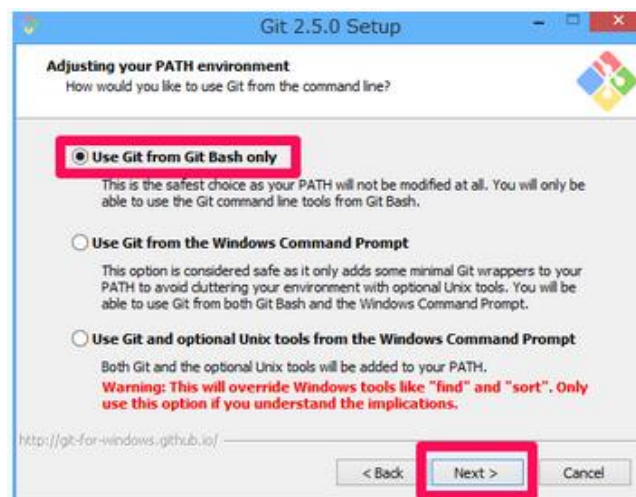
Step2 次にダウンロードしたファイルをダブルクリックします。



Step3 [Next >] ボタンをクリックします。

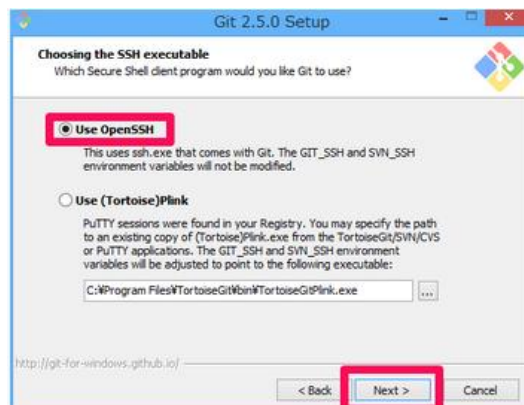


Step4 [Use Git Bash only] を選択し、[Next >] ボタンをクリックします。

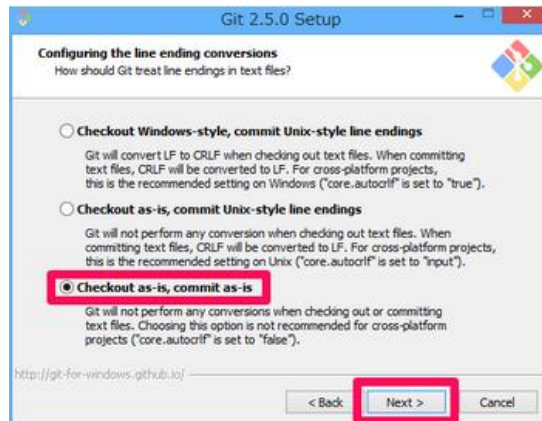


\* [Use git from the windows command prompt] はコマンドプロンプトで git を使う場合に選択します

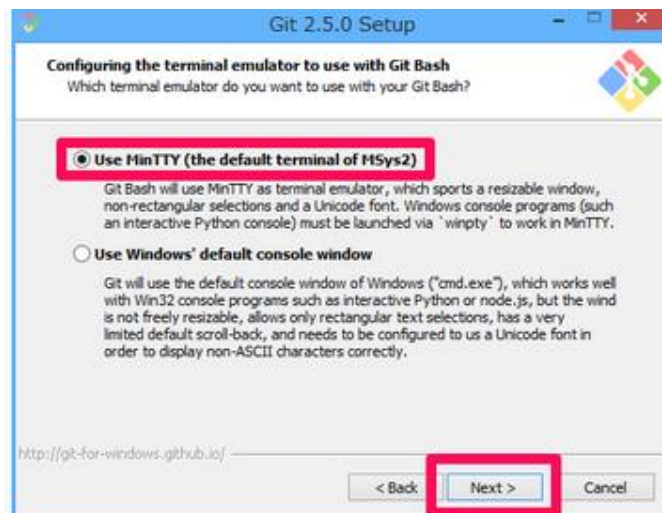
Step5 [Use OpenSSH] を選択します。



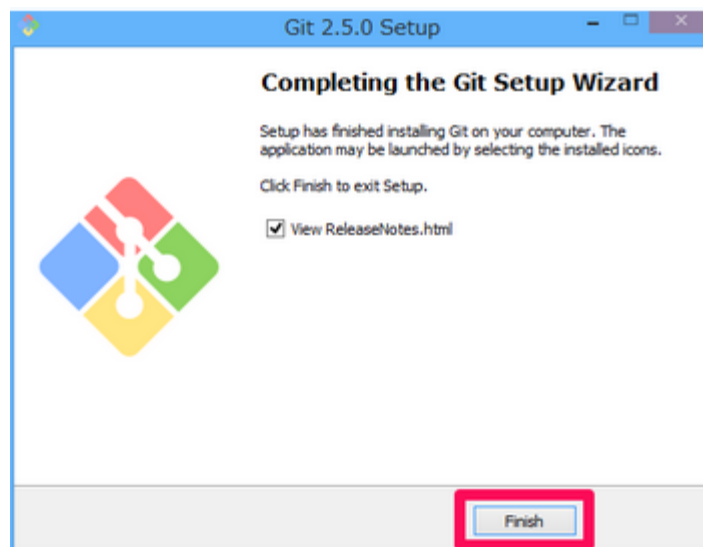
Step6 [Checkout as-is, commit as-is] を選択します。



Step7 [Use MinTTY (the default terminal of Msys2)] を選択します。



Step8 [Finish] ボタンをクリックします。

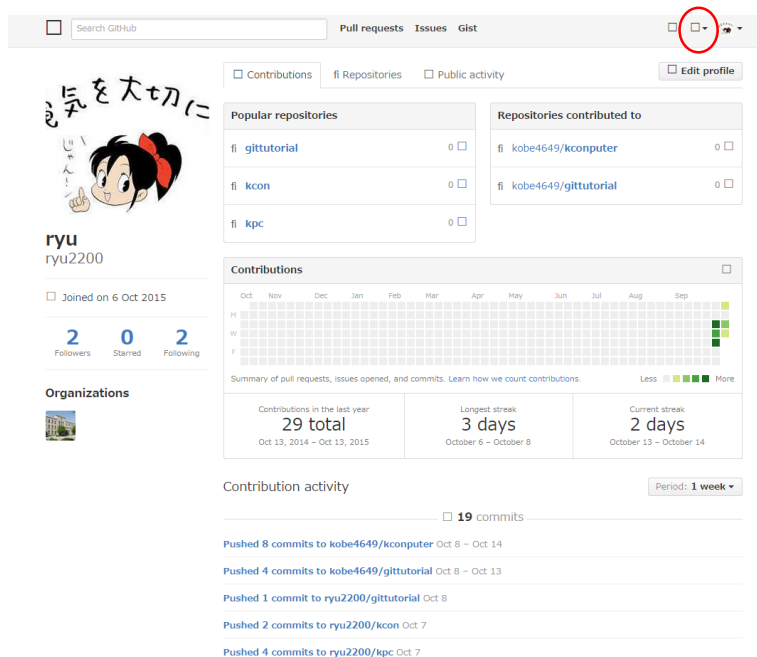


### 3. Git の基本的操作

ここでは git の基本操作について説明します。なお git は本来チームで使う分散型バージョンシステムですがここでは簡単として「個人」で複数台の pc を用い開発する場合を想定し git の基本操作を説明します。

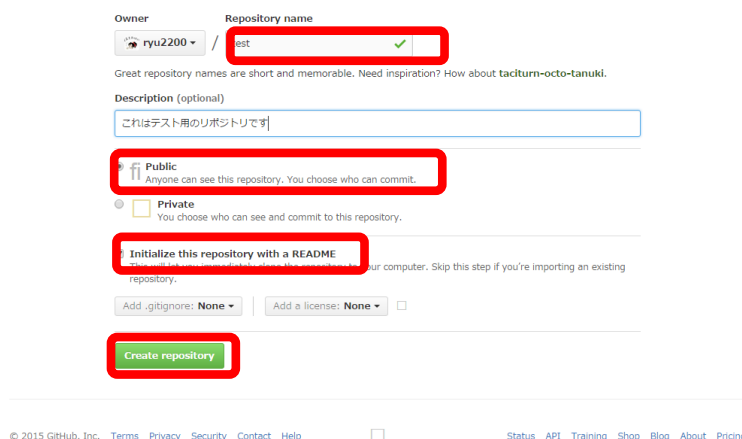
#### Step1 リモートリポジトリの作成と初期化

Github にログインし赤丸をクリックし表示されるメニューの「NewRepository」をクリックします



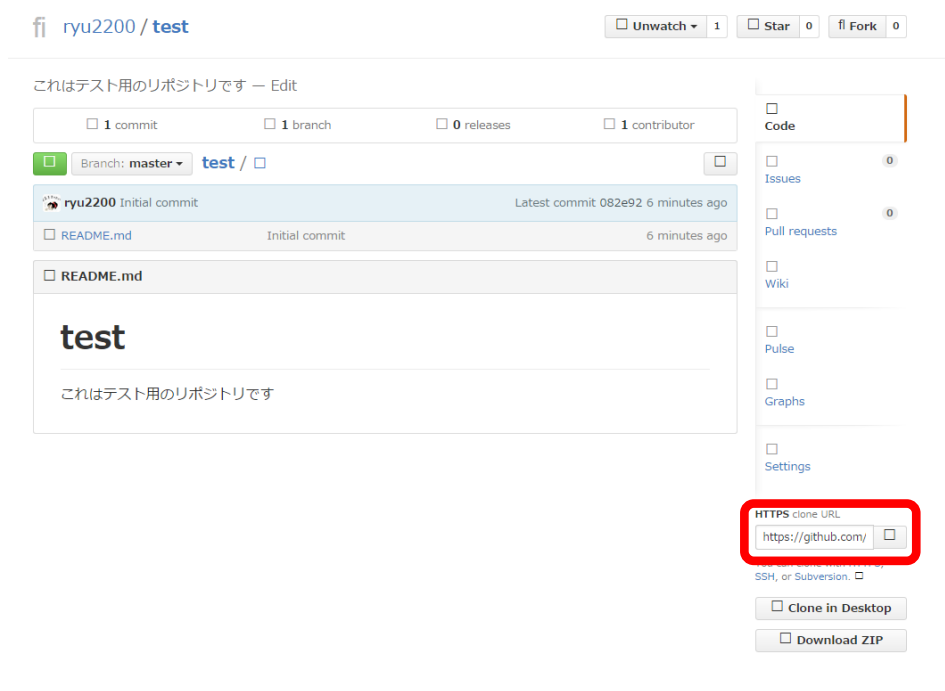
図のような画面が出ますので「repository name」に好きなレポジトリ名を入力します。次に[public]と「Initialize this repository with a README」にチェックを入れリポジトリを初期化します。

最後に「Create Repository」をクリックし新規リモートリポジトリを作成します。



\*Public リポジトリは他者が自由に編集することはできませんが **github** 上に公開されてしまいます。公開したくないファイルがある場合 **private** リポジトリ(月 7 \$ 以上の有料会員では無いと選べません)を選択してください。

すると次の様な画面が出てきます



\*[HTTPS clone URL]はこのリポジトリにアクセスするための URL です。後々リモートリポジトリの URL が必要となったら、この URL をコピーすると便利です。

## Step2 初期設定

手元の pc で **git** を使うためにはサーバーにログインするための情報を予め登録しておく必要があります。今回使用するサーバーは **github** サーバーです。

ここで必要となるのが先ほど作成した **github** アカウントのユーザー名と **e-mail** アドレスとパスワードです。まずデスクトップ上の **Git Bash** をクリックします。



すると **MinGW** と表示されるシェルが出てきます。次に **MinGW** 上で

```
git config --global user.names <github ユーザー名>
```

```
git config --global user.email <github メールアドレス>
```

と打ち込んでください。

```
MINGW64:/c/Users/iryolabo
iryolabo@iryolabo-PC MINGW64 ~
$ git config --global user.names ryu2200
iryolabo@iryolabo-PC MINGW64 ~
$ git config --global user.email dragonash2200@gmail.com
iryolabo@iryolabo-PC MINGW64 ~
$ |
```

### Step3 リモートリポジトリの複製

次に `git clone https://github.com/ユーザー名/リポジトリ名.git` と打ち自身の pc 上にリモートリポジトリと全く同じ内容のローカルリポジトリを作成します。

```
MINGW64:/c/Users/iryolabo
--template <template-directory>    directory from which templates will be used
--reference <repo>                  reference repository
--dissociate                         use --reference only while cloning
-o, --origin <name>                 use <name> instead of 'origin' to track upstream
-b, --branch <branch>               checkout <branch> instead of the remote's HEAD
-u, --upload-pack <path>             path to git-upload-pack on the remote
--depth <depth>                     create a shallow clone of that depth
--single-branch                     clone only one branch, HEAD or --branch
--separate-git-dir <gitdir>         separate git dir from working tree
-c, --config <key=value>            set config inside the new repository

iryolabo@iryolabo-PC MINGW64 ~
$ git clone https://github.com/ryu2200/test.git
Cloning into 'test'...
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.
iryolabo@iryolabo-PC MINGW64 ~
$ |
```

### Step4 ファイルのステージングへの登録

Git ではステージという概念があります。ステージはファイルの変更履歴を「commit」記憶する前の準備段階の領域です。

この「Commit」とはファイルを追加したり変更したりローカルリポジトリの状態が変化した時、その変更した状態を履歴として記憶する操作のことで、これを行うことで git は簡単にバージョン管理が行えるようになっています。ここではコミットする前の段階ステ

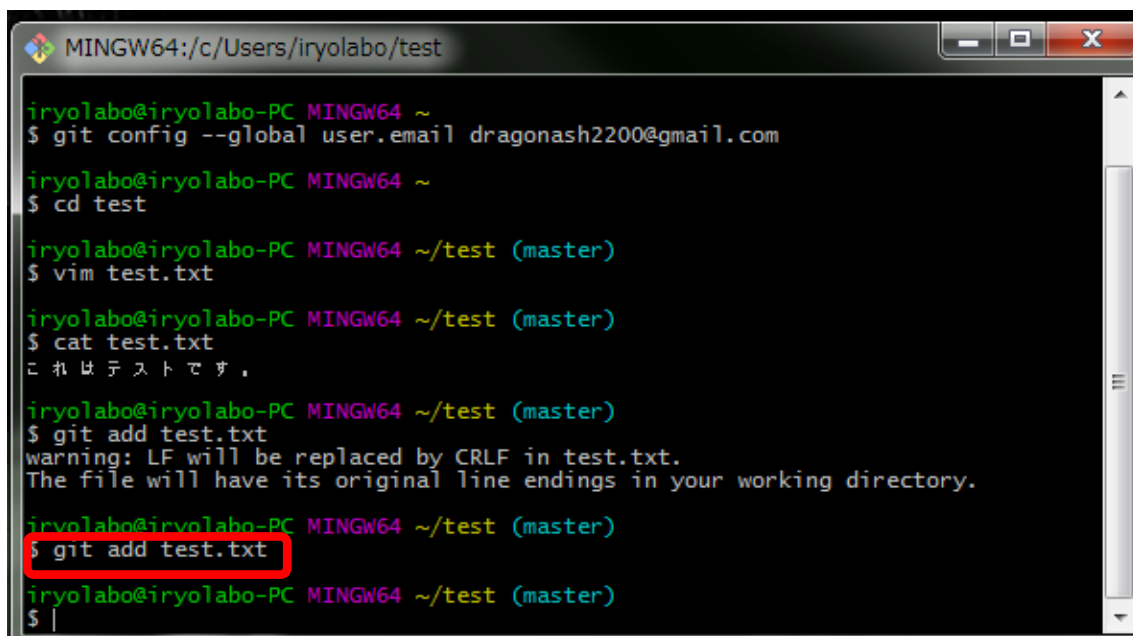


ージに登録する方法について説明します。

まず

`git add <変更したファイル,または追加するファイル>`

と打ち変更したファイルをステージに登録します。

A terminal window titled 'MINGW64:/c/Users/iryolabo/test' showing a series of commands and their outputs. The commands include setting the global user email, changing to the 'test' directory, editing 'test.txt' with vim, viewing it with cat, and adding it to the staging area with git add. A warning about line endings is shown. The final command 'git add test.txt' is highlighted with a red rectangle.

```
MINGW64:/c/Users/iryolabo/test

iryolabo@iryolabo-PC MINGW64 ~
$ git config --global user.email dragonash2200@gmail.com

iryolabo@iryolabo-PC MINGW64 ~
$ cd test

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ vim test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ cat test.txt
これはテストです。

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git add test.txt
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git add test.txt

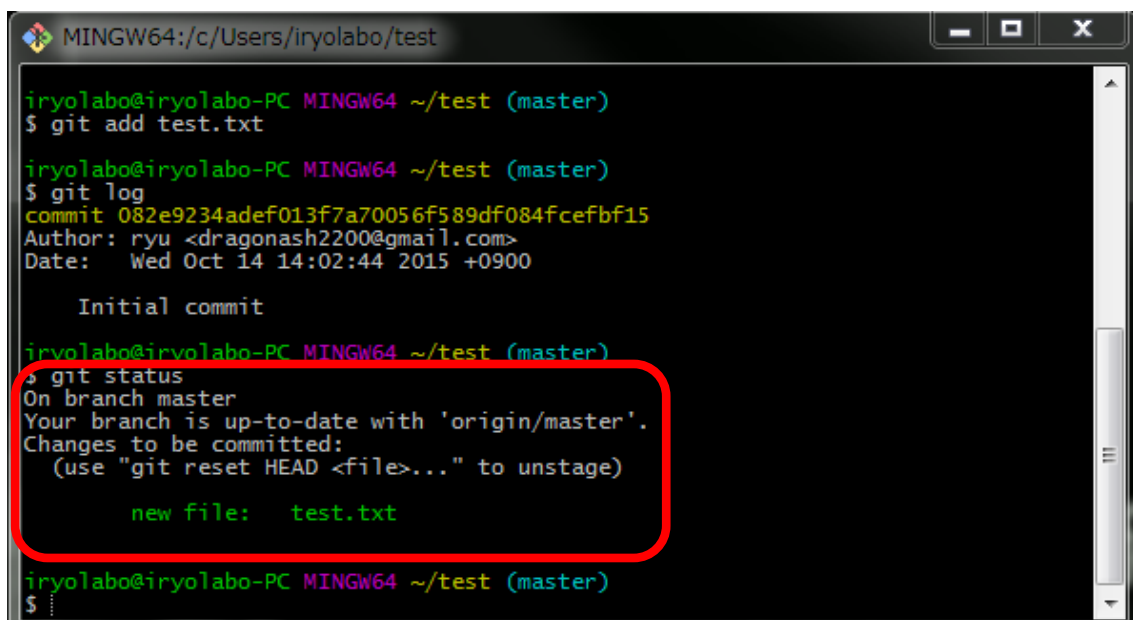
iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ |
```

するとステージに登録したファイルは tracked ファイル(追跡対象ファイル)となります。

また追跡対象ファイルは

`git status`

で確認できます

A terminal window titled 'MINGW64:/c/Users/iryolabo/test' showing the output of 'git log' and 'git status'. The 'git log' output shows an initial commit. The 'git status' output shows that the branch is up-to-date and that 'test.txt' is a new file to be committed. The 'git status' output is highlighted with a red rectangle.

```
MINGW64:/c/Users/iryolabo/test

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git add test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git log
commit 082e9234a7ef013f7a70056f589df084fcefbbf15
Author: ryu <dragonash2200@gmail.com>
Date:   Wed Oct 14 14:02:44 2015 +0900

    Initial commit

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ |
```

ステージングから外したいときは

git reset

とうちます。

すると次のようにステージに加えたファイルが非追跡対象となりステージから除外されている事がわかります。

```
MINGW64:/c/Users/iryolabo/test
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git reset

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    test.txt

nothing added to commit but untracked files present (use "git add" to track)
iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

Step4 ステージに追加したファイルをコミットする

Step3 でステージに登録した追跡対象ファイルを

git commit -m"コメント文"

と打ち commit します。

```
MINGW64:/c/Users/iryolabo/test
  (use "git add <file>..." to include in what will be committed)

    test.txt

nothing added to commit but untracked files present (use "git add" to track)

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git add test.txt
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git add test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git commit -m"テスト用ファイルを追加しました"
[master 951b8ad] テスト用ファイルを追加しました
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.
1 file changed, 1 insertion(+)
create mode 100644 test.txt

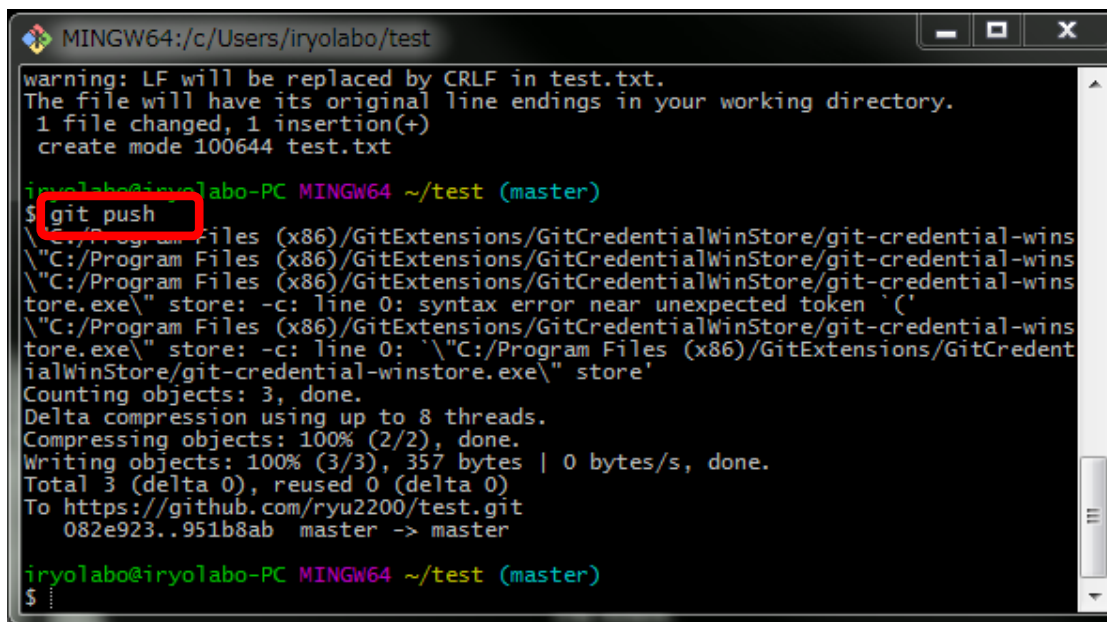
iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

Step5 コミット履歴を push する

ローカルリポジトリのコミット履歴を

git push

と打ちリモートリポジトリに反映させます。

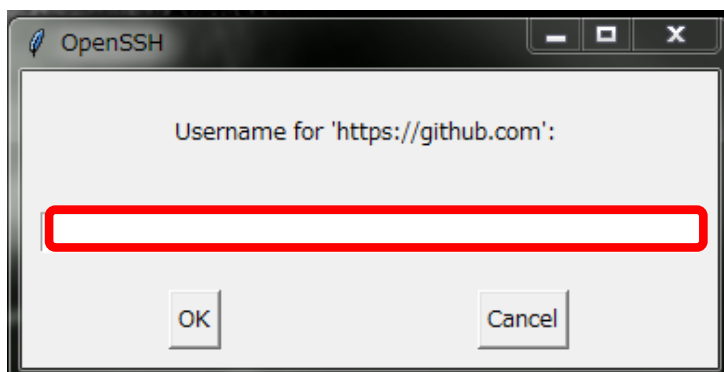


```
MINGW64:/c/Users/iryolabo/test
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.
1 file changed, 1 insertion(+)
create mode 100644 test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git push
\ "C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
\C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
\C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
tore.exe\" store: -c: line 0: syntax error near unexpected token '('
\C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
tore.exe\" store: -c: line 0: \"C:/Program Files (x86)/GitExtensions/GitCredent
ialWinStore/git-credential-wins\" store'
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 357 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ryu2200/test.git
082e923..951b8ab master -> master

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

次に Push するリモートリポジトリのユーザー名の確認が出るので入力します

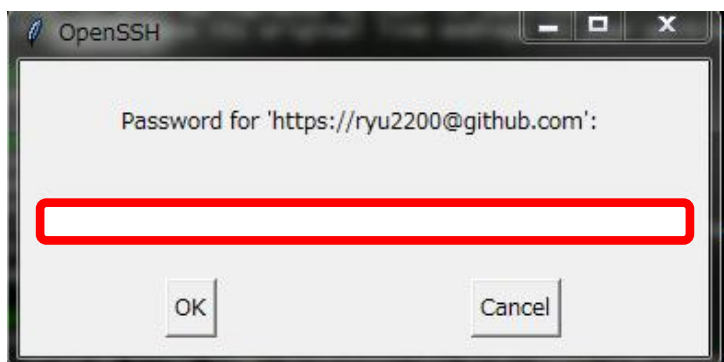


OpenSSH

Username for 'https://github.com':

OK Cancel

パスワードを入力します



OpenSSH

Password for 'https://ryu2200@github.com':

OK Cancel

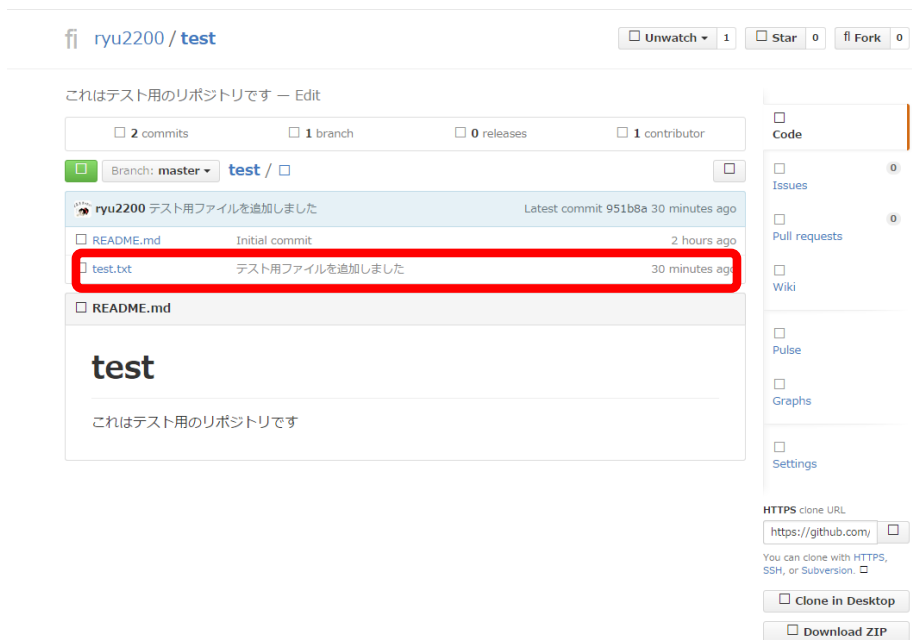
すると次の様になりリモートリポジトリへのコミットの登録が終了します。

```
MINGW64:/c/Users/iryolabo/test
warning: LF will be replaced by CRLF in test.txt.
The file will have its original line endings in your working directory.
1 file changed, 1 insertion(+)
create mode 100644 test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git push
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
tore.exe\" store: -c: line 0: syntax error near unexpected token '('
"C:/Program Files (x86)/GitExtensions/GitCredentialWinStore/git-credential-wins
tore.exe\" store: -c: line 0: \"C:/Program Files (x86)/GitExtensions/GitCredent
ialWinStore/git-credential-winstore.exe\" store'
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 357 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ryu2200/test.git
082e923..951b8ab master -> master

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

すると次のようにリモートリポジトリにファイルが追加されたことがわかります。



## Step 6 pull

他の pc や他者とチーム開発をする際、自身の知らないうちにリモートリポジトリにコミットが追加されている場合があります。その際 **push** しリモートリポジトリを変更する前に **pull** しリモートリポジトリの最新の履歴をローカルリポジトリに取得する必要があります。

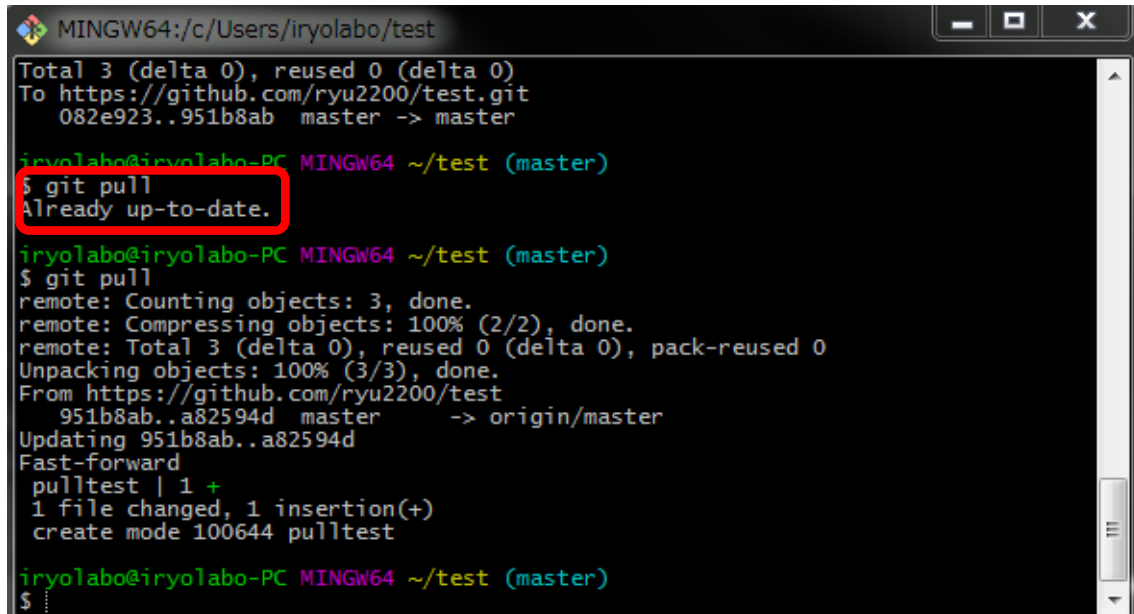
もしこれが行われないとローカルリポジトリとリモートリポジトリは常に同一の状態にあるので、あなたが **push** した時点で他の人が **push** してリモートリポジトリに反映させた

履歴は知らないうちに消えてしまう事になります。

そのためまず

`git pull`

とうち、図のようにリモートリポジトリの最新の履歴を取得してから `push` してください。

A terminal window titled 'MINGW64:/c/Users/iryolabo/test' showing the execution of 'git pull'. The first command results in 'Already up-to-date.' which is highlighted with a red box. The second command shows a successful pull from the remote repository, updating the local master branch to the latest state (a82594d) and creating a new file 'pulltest.txt'.

```
MINGW64:/c/Users/iryolabo/test
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ryu2200/test.git
   082e923..951b8ab  master -> master

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git pull
Already up-to-date.

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git pull
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/ryu2200/test
   951b8ab..a82594d  master      -> origin/master
Updating 951b8ab..a82594d
Fast-forward
 pulltest | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 pulltest

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

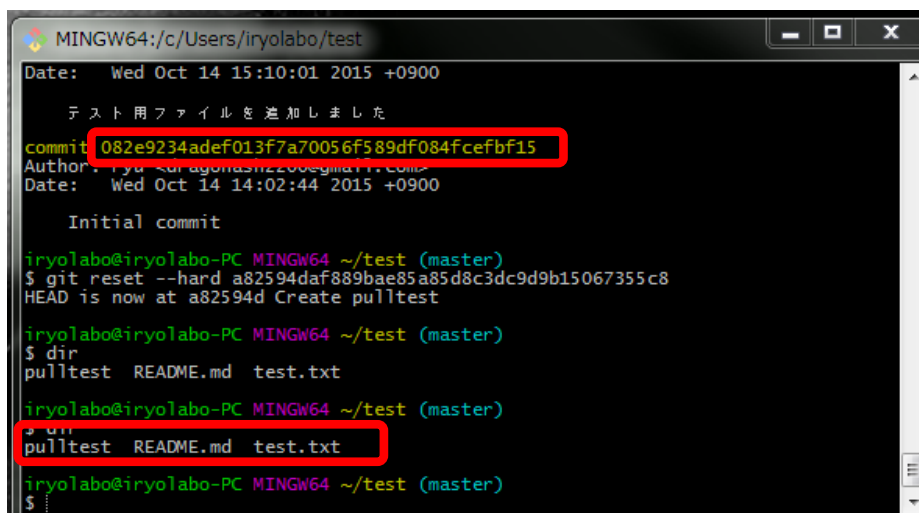
補足 過去の状態に戻る `log+reset`

Git では `commit` 時のハッシュ値を使って過去の状態に戻す事ができます。

まず

`git log`

と打ち画面の様に `commit` のハッシュ値を取得します

A terminal window titled 'MINGW64:/c/Users/iryolabo/test' showing the execution of 'git log'. The output displays the commit history, with the first commit (082e9234) highlighted by a red box. Below this, the user performs a 'git reset --hard' to the specified commit, followed by a 'git log' command which shows the current state (pulltest) and the files 'README.md' and 'test.txt', with the command itself highlighted by a red box.

```
MINGW64:/c/Users/iryolabo/test
Date:   Wed Oct 14 15:10:01 2015 +0900

テスト用ファイルを追加しました
commit 082e9234def013f7a70056f589df084fcefbbf15
Author: iryolabo <iryolabo@ryu2200egmail.com>
Date:   Wed Oct 14 14:02:44 2015 +0900

Initial commit

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git reset --hard a82594daf889bae85a85d8c3dc9d9b15067355c8
HEAD is now at a82594d Create pulltest

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ dir
pulltest README.md test.txt

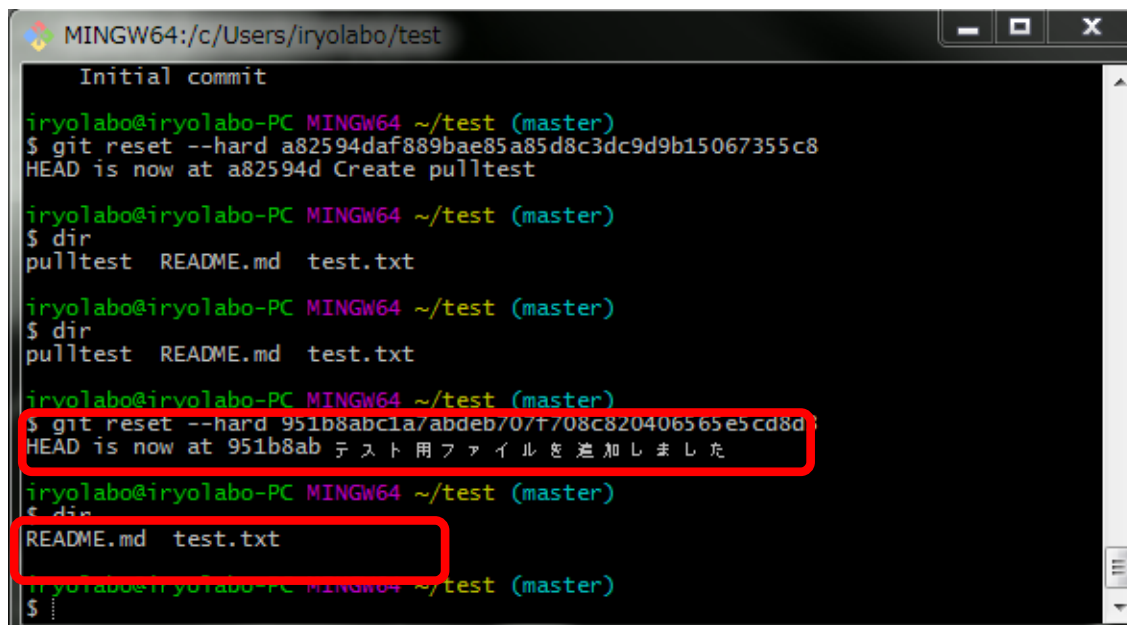
iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git log
pulltest README.md test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

次に

`git reset --hard <ハッシュ値>`

とします。



```
MINGW64:/c/Users/iryolabo/test
Initial commit
iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git reset --hard a82594daf889bae85a85d8c3dc9d9b15067355c8
HEAD is now at a82594d Create pulltest

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ dir
pulltest  README.md  test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ dir
pulltest  README.md  test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ git reset --hard 951b8abc1a7abdeb707f708c820406565e5cd8d
HEAD is now at 951b8ab テスト用ファイルを追加しました

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$ dir
README.md  test.txt

iryolabo@iryolabo-PC MINGW64 ~/test (master)
$
```

すると画面の様に過去のコミット時の状態にローカルリポジトリが変更されていることがわかります。