



ハンズオンタイム1



Macの人は**ターミナル**、windowsの人は**Gitbash**を準備してください。

ここからは、前回学んだgitの基本的操作を実際にやってみます！

前回あまり理解できなかった人でも実際に手を動かすことで、理解できると思います。

自分はMacのため、ターミナルの画面で説明していきます。

Windowsの人も基本的には同じなので、一緒に進めていってください。

グレーで囲まれているところをターミナル、またはGitbashで入力してください。

まず、gitを使うための設定をしていきます。最初に以下のコマンドで確認してください。



```
git config --list
```

いくつか情報が書かれていると思いますが、最低以下の2つが書かれてあれば大丈夫です。

「user.name」は、自分のgitアカウントのユーザーネームです。

「user.email」は、gitアカウントを作成した際に使ったメールアドレスです。基本、大学のかな？



```
user.name="gitのユーザーネーム"
```

```
user.email="gitで設定したメールアドレス"
```

2つの情報が設定されていなかった人は、以下のコマンドをそれぞれ実行してください。

```
▶ git config --global user.name "gitのユーザーネーム"
```

```
▶ git config --global user.email "gitで設定したメールアドレス"
```

最後にもう一度、「git config --list」で確認し、入力したユーザーネームとメールアドレスが確認できたら設定終了です。

ここから実際に操作していきます。

ターミナルに慣れている人は、任意のディレクトリで進めてください。

慣れていない人は、以下で進めてください。これは、ホームディレクトリに移動するよという意味です。「pwd」で確認してみてください。

```
▶ cd ~
```

次に、ホームディレクトリに新たに「git-practice」というディレクトリを作ります。

```
▶ mkdir git-practice
```

作成したディレクトリに入ります。

```
▶ cd git-practice
```

「git-practice」をワークツリーとし、この中にgitディレクトリを作ります。
前回学んだ、「init」を使います。

▶ `git init`

以下のように、「.git」という隠しディレクトリができていれば大丈夫です。
この隠しディレクトリが自分たちの歴史を管理するために使われるものです。

👉 Initialized empty Git repository in /Users/toriyabekako/git-practice/.git/

今回は、練習のため自己紹介のテキストファイルgitで管理してみましょう！
以下のコマンドを入力してください。

▶ `touch self-introduction.txt`

Macの人はFinder、Windowsの人は、エクスプローラーを開き「git-practice」の中に「self-introduction.txt」があることを確認してください。

その「self-introduction.txt」を開き、1行何か追加して保存してください。

できたら、閉じてOKです。

例)



前回の説明にはなかった便利なコマンドをここで使います。

▶ `git status`

以下のようなレスポンスが返ってくると思います。



On branch main

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

`self-introduction.txt`

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

このようにステータスを見ると、まだこのファイルは作成してから、gitの管理下に置かれていません（コミットされていない）ということが読み取れます。そのため、コミットをしていきましょう。コミットする前に、ステージ済みにするためにインデックスに上げる必要があるということを前回学んだはずです。そのためには、「add」する必要があります。

```
▶ git add self-introduction.txt
```

その後、またステータスを確認してみてください。

```
▶ git status
```

次は、以下のレスポンスが返ってきたと思います。

```
👉 On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   self-introduction.txt
```

1つ前のステータスでは、「self-introduction.txt」が「Untracked files」の中に入っていたと思います。しかし、今回は、「Changes to be committed」に入っています。これは、コミットする準備ができていますよという意味です。(ステージ済みという意味)

コミットの準備ができたため、早速コミットしてみましょう。

```
▶ git commit -m "Initial commit"
```

""の間には、コメントを入れることができます。コメントを入れる時は、「-m」をつけます。

今は、「self-introduction.txt」にプロフィールを追加したということをコミットで記録しようとしています。それと同時に、コミットには、いろいろな記録が付与されます。例えば、誰が記録したか、いつ記録したかなどの情報です。その中の1つに、gitが提供している**コミットコメント**というものがあります。これは、このコミットがどのようなことをしているかを一言で表す機能です。

今回は、最初のコミットという意味の"Initial commit"を付与します。

では、またステータスを確認してみましょう。

▶ git status

以下のようなレスポンスが返ってくると思います。

👉 On branch main
nothing to commit, working tree clean

これは、最後にコミットしてから、ワーキングツリーになんの変更もありませんということが書いてあります。これで全てのファイルがコミット済みということが確認できました。

ここまで、常にステータスを確認してきましたが、git操作になれるまでは、何かと操作をするたび確認することをおすすめします！状況の確認を行えるので、苦手な人は、確認しながら進めてみてください！

次に、コミットしたファイルのどのような情報が付与されたのかを実際に確認してみましょう。

▶ git log

以下のようなレスポンスが返ってくると思います。先ほど、説明した通り、誰が、いつ、どのようなコミットをしたのかが読み取れるはずです。一行目は、コミットを

一意に特定するためのハッシュ値になります。基本的に違うコミットであればハッシュ値は被りません。

今はコミットを一回しかしていないため、1回分の情報しかありませんが、コミットを何回か行っていくうちに、数は増えていきます。



`commit a2cba7ae1a1bfb45026f6ceefbc144082c6f2cf7 (HEAD → main)`

Author: b2211700 <b2211700@photon.chitose.ac.jp>

Date: Sat Oct 21 14:56:58 2023 +0900

Initial commit

これで、ローカルリポジトリでの基本的な操作はできているということになります。

あとは、回数を重ねて慣れていきましょう！