

Gitはオープンソースとして配布されているバージョン管理システムです。Gitを使うと、あなたのノートまたはデスクトップパソコンから、GitHub上のアクティビティーを操作できます。この早見表にはコマンドラインからよく使われているGitの命令のまとめています。

GITのインストール

GitHubは、利用頻度の高いリポジトリへのアクションを可能にするGUI版と、上級向けに自動的にアップデートされるGitのコマンドライン版を含むデスクトップクライアントを提供しています。

GitHub for Windows

<https://windows.github.com>

GitHub for Mac

<https://mac.github.com>

LinuxまたはPOSIXシステムのためのGitディストリビューションは公式のGit SCMウェブサイトから入手できます。

Git 全プラットフォーム版

<http://git-scm.com>

ツールの設定

すべてのローカルリポジトリ用の、ユーザー情報設定方法

```
$ git config --global user.name "[name]"
```

コミット操作に付加されるあなたの名前を設定します

```
$ git config --global user.email "[email address]"
```

コミット操作に付加されるあなたのメールアドレスを設定します

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

コマンドラインの出力を見やすくするため色を設定できます

リポジトリの作成

リポジトリを新規作成するもしくは既存のURLから取得します

```
$ git init [project-name]
```

指定した名前のローカルリポジトリを作成します

```
$ git clone [url]
```

プロジェクトとすべてのバージョン履歴をダウンロードします

変更の作成

変更をレビューしコミット操作ログを作成します

```
$ git status
```

コミット可能なすべての新規または変更のあるファイルを一覧で表示します

```
$ git diff
```

まだステージされていないファイルの差分を表示します

```
$ git add [file]
```

バージョン管理のためにファイルのスナップショットを作成します

```
$ git diff --staged
```

ステージングと最後のファイルバージョンとの差分を表示します

```
$ git reset [file]
```

ファイルをステージングから外しますが、その内容は保持します

```
$ git commit -m "[descriptive message]"
```

ファイルのスナップショットをバージョン履歴内に恒久的に記録します

変更の整理

一連のコミットに名前をつけ、完了した成果を結合します

```
$ git branch
```

現在のリポジトリ上のすべてのローカルブランチを一覧で表示します

```
$ git branch [branch-name]
```

新規ブランチを作成します

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

指定されたブランチに切り替え、作業ディレクトリを更新します

```
$ git merge [branch]
```

指定されたブランチの履歴を現在のブランチに統合します

```
$ git branch -d [branch-name]
```

指定されたブランチを削除します

GIT チートシート

ファイル名の整理

バージョン管理されているファイルの移動、または削除を行ないます

```
$ git rm [file]
```

作業ディレクトリからファイルを削除し、削除をステージします

```
$ git rm --cached [file]
```

バージョン管理からファイルを削除して、ローカルのファイルは保持します

```
$ git mv [file-original] [file-renamed]
```

ファイル名を変更し、コミットします

トラッキングの制限

一時ファイルやパスを除外します

```
*.log  
build/  
temp-*
```

.gitignore という名前のテキストファイルで、指定されたパターンに該当するファイルやパスを誤ってバージョン管理してしまうことを防げます

```
$ git ls-files --other --ignored --exclude-standard
```

プロジェクト内のすべての除外されたファイルを一覧で表示します

断片の保存

未完成の変更を一時的に退避し、復旧させることができます

```
$ git stash
```

すべての変更のあるトラックされているファイルを一時的に保存します

```
$ git stash pop
```

直近に一時保存されたファイルを復旧します

```
$ git stash list
```

すべての一時保存された変更セットを一覧で表示します

```
$ git stash drop
```

直近に一時保存された変更セットを破棄します

履歴の確認

プロジェクトファイルの進展を確認します

```
$ git log
```

現在のブランチのバージョン履歴を一覧で表示します

```
$ git log --follow [file]
```

名前の変更を含む指定したファイルのバージョン履歴の一覧を表示します

```
$ git diff [first-branch]...[second-branch]
```

2つのブランチ間の差分を表示します

```
$ git show [commit]
```

指定されたコミットのメタ情報と変更内容を出力します

コミットの修正

ミスの削除と履歴の置き換え

```
$ git reset [commit]
```

[commit] 以降すべてのコミットを取り消し、ローカルでは変更を保持します

```
$ git reset --hard [commit]
```

指定されたコミットに戻り、それ以降のすべての変更を破棄します

変更の同期

リポジトリのブックマークを登録し、バージョン履歴を交換します

```
$ git fetch [bookmark]
```

リポジトリブックマークからすべての履歴をダウンロードします

```
$ git merge [bookmark]/[branch]
```

ブックマークのブランチを現在のローカルブランチに統合します

```
$ git push [alias] [branch]
```

すべてのローカルブランチのコミットをGitHubにアップロードします

```
$ git pull
```

ブックマークの履歴をダウンロードし、変更を統合します

GitHub Training

より詳しく GitHub と Git の使い方を知るには、トレーニングチームにメールするか、イベントスケジュールや受講可能なプライベート講義を知るために私たちのウェブサイトをご覧ください。

✉ training@github.com
🌐 training.github.com