

# SourceTreeによるGit 入門

プログラマーからデザイナーまで  
主婦でも使えるファイルのバージョン管理

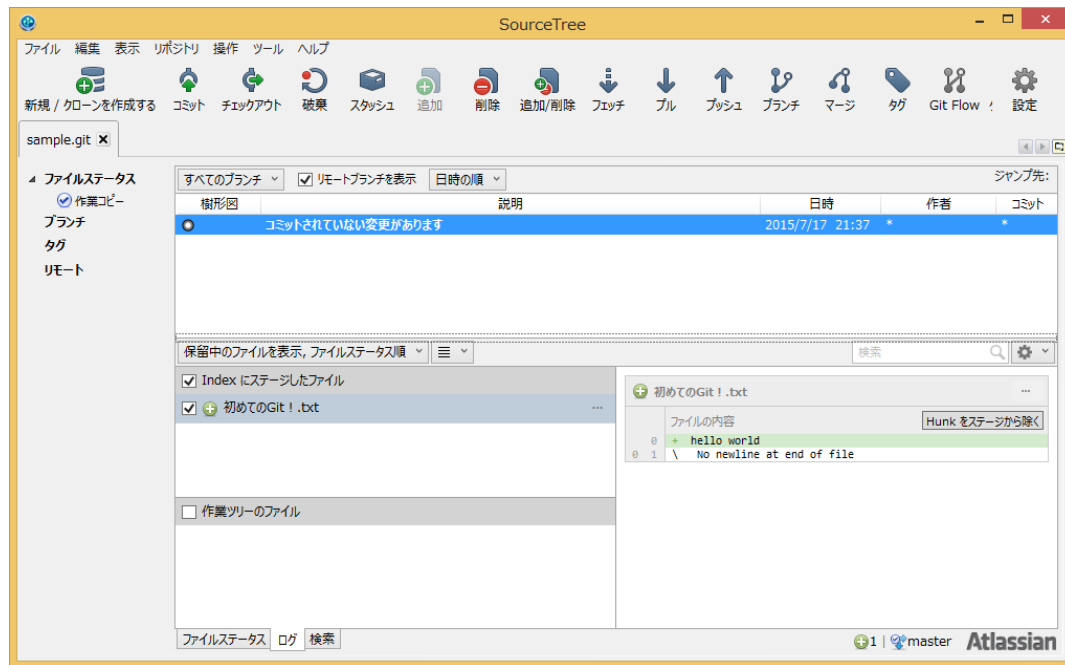
# えっえっ、Gitってなんですか？

- Gitとは？  
分散型バージョン管理システムである…！
- バージョン管理？  
ファイルの変更履歴を記録して、いつでも差分を確認したり、前の状態に戻したりできる！
- Git以外にもバージョン管理システムはある！  
Mercurial、Subversion(集中型バージョン管理)など

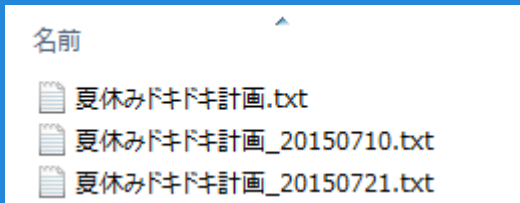
# はて、SourceTreeとな？

Gitの操作を画面で  
簡単に操作できる！

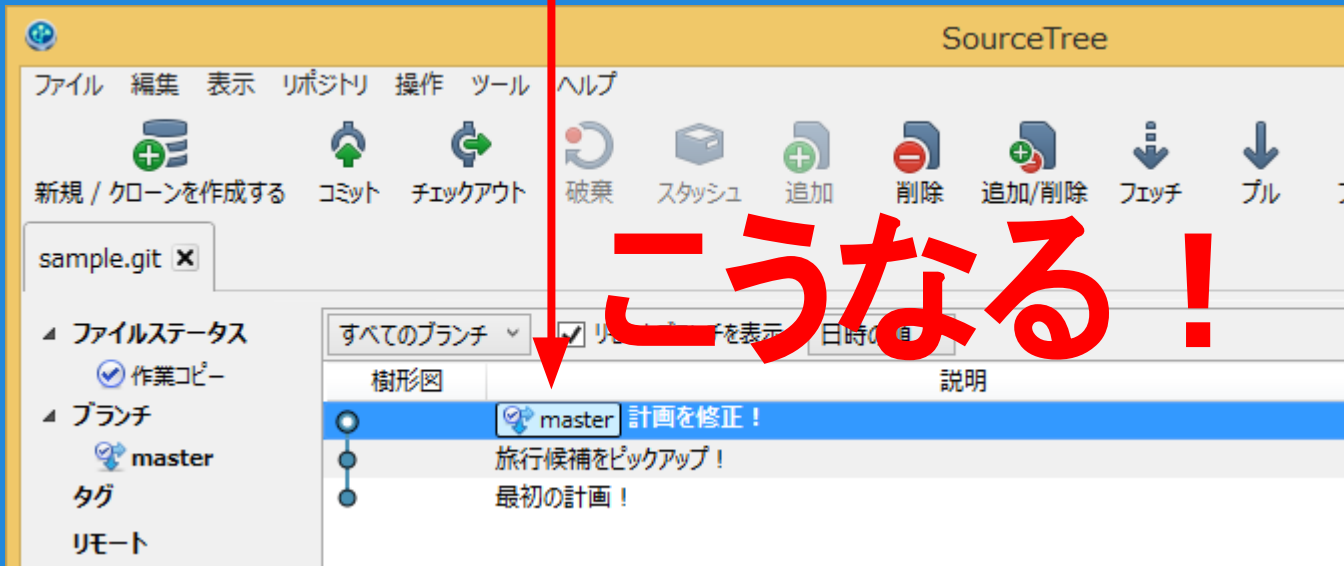
基本的にGitは  
コマンドで  
操作するもの！



# つまり!!



これが…



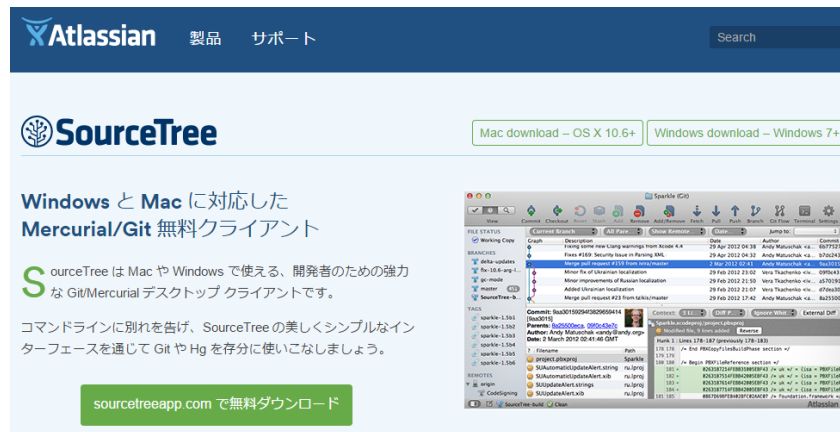
変更履歴や差分が見れる！自由に前の状態に戻せる！

# SourceTreeのインストール

## ダウンロードしてインストールしましょう！

### Gitのインストールとバージョン

- Mac  
初期でGitがインストールされているが、バージョンが古い。
- Windows  
SourceTree内蔵のGitが使える。  
正式には、Git for Windowsと検索してみよう。

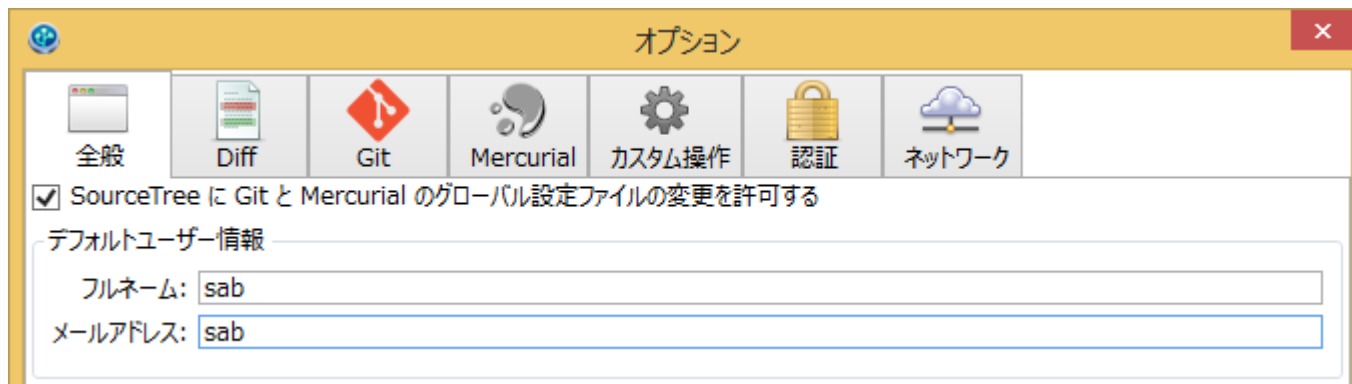


The image shows the SourceTree website interface. At the top, there's an Atlassian logo and navigation links for '製品' (Products) and 'サポート' (Support). Below that, the SourceTree logo is prominent. There are two buttons for downloading: 'Mac download - OS X 10.6+' and 'Windows download - Windows 7+'. The main content area features the text 'Windows と Mac に対応した Mercurial/Git 無料クライアント' (Windows and Mac compatible free Mercurial/Git client). A green button at the bottom says 'sourcetreeapp.com で無料ダウンロード' (Free download at sourcetreeapp.com). On the right side, there's a screenshot of a terminal window showing a Git commit message: 'Merge pull request #23 from takis/master'.

# SourceTreeの初期設定

SourceTreeを起動したら、ユーザー情報を入れておこう。メールアドレスは適当でもOK。

- ・Mac: SourceTree → 環境設定
- ・Windows: ツール → オプション

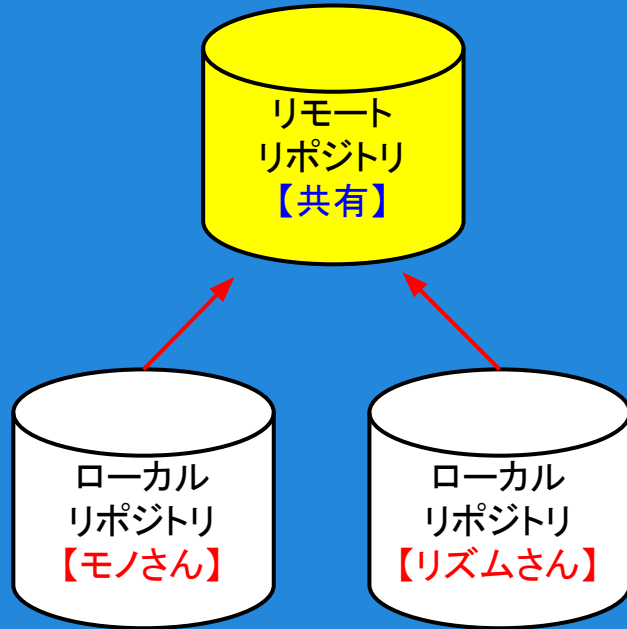


フルネームの名前は、ファイルのコミット者として変更履歴に記録されます。

# Gitを使う準備

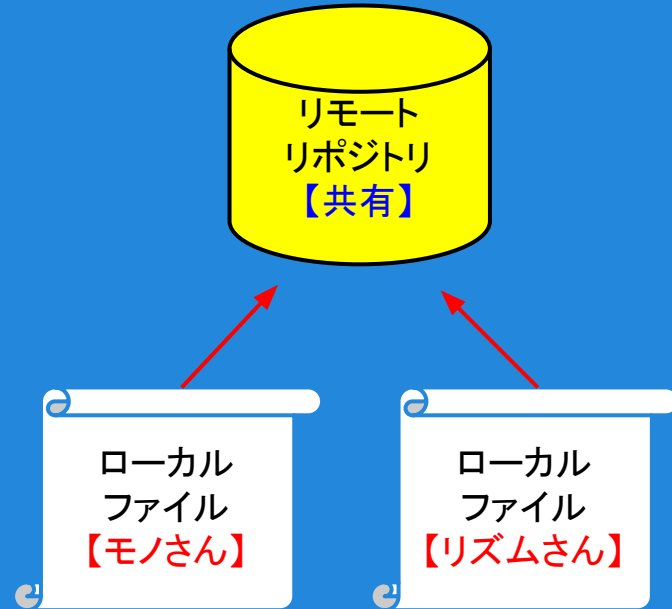
- リポジトリ … 変更履歴を管理する倉庫
  - ローカルリポジトリ (ノンベアリポジトリ)  
好きに使っていい自分のリポジトリ。
  - リモートリポジトリ (ベアリポジトリ)  
サーバーにあって皆で共有するリポジトリ。
    - GitHubやBitBucketなどリポジトリを提供するWebサービスがある。
    - SourceTreeはローカル操作が主なので、リモートリポジトリをローカル内に作るにはコマンドが必要になる！ `git init --bare --shared`

# 分散型



ローカルにリポジトリがあるので  
いつでも変更履歴が見れる！

# 集中型



リポジトリを見るにはネットワークが必須／ (^o^)\

分散型と集中型のリモートリポジトリとローカルリポジトリ



# 実演編 Gitの基本操作！

## 送信系

- コミット … 作業完了！変更履歴に記録！
- プッシュ … リモートへアップロード！

## 受信系

- クローン … リモートからローカルに新規で全コピー！
- フェッチ … リモートの最新状態をダウンロード！
- マージ …… 別のコミットを自分にインストール！
- プル …… フェッチとマージを同時に行う操作！

# 実演編 Gitの便利操作！

- リセット … 駄作は没！セーブポイントからやり直し！
- アmend … やっぱこっち！直前のコミットを修正！
  - プッシュした後は「やっぱこっち！」は禁止！
  - リモートは皆と共有しているので「間違えました、修正コミットします！」と新しくコミットする。
  - アmendのような歴史操作はローカル内のみ厳守！
- ブランチ … 分岐して並行作業！別案や新機能を作成！

# 設定編 Gitの設定

- **.gitignore … 部外者だ！無視ファイル！**
  - ログやテンプファイルなどコミット対象から除外。
  - <https://www.gitignore.io/>
  - <https://github.com/github/gitignore>
- **.gitkeep … 空フォルダもコミットしたい！**
- **.gitconfig … 設定ファイル！**
  - ignorecaseは、false推奨。(文字の大小を区別なし)
  - 外部Diff、マージツールを設定できる。(WinMergeおすすめ)

# 運用編 Gitのブランチモデル

- git-flow … 定期的なリリース向け！
  - master … リリース用、タグ用。
  - hotfix … リリース済みの緊急修正用。
  - release … リリース準備用。
  - develop … メイン開発用、常にビルド可。
  - feature … 個別機能やバグ修正の開発用。
- GitHub Flow … 頻繁なバージョンアップ向け！
  - master … 常にリリース可能。
  - add,fix … 小さなブランチで素早い開発。

# おしまい

まずは一人で使って簡単な操作から慣れてみよう！  
そして、Gitはまだまだたくさんの事ができるよ！すごいね！  
(コミット時に自動処理、ユーザーごとのブランチ権限など)