



**G's ACADEMY**  
**TOKYO**

# GitHub入門 (2015.12.26)

azm



# GitHub入門 アウトライン

1. バージョン管理システムってなんだろう
2. Git とは？
3. 良く聞く GitHub って？
4. GitHub の画面説明
5. Git クライアント SourceTree の紹介と説明
6. Git にコミットしてみよう！
7. GitHub にプッシュしてみよう！

# バージョン管理システムがないと

index.php

index\_20151225.php

index\_最新.php



どれが最新版？

# バージョン管理システムがあると

index.php

最新版は  
これ！

History for gscms / index.php

Commits on Jul 9, 2015	
 データベース接続情報を共通ファイル化 azgz authored 4 days ago	 4483c78 
Commits on Jul 2, 2015	
 update.php へのリンクを実装 azgz authored 11 days ago	 82f778f 
 update.php への遷移指示を追加 (htmlの構造がおかしいのも修正) azgz authored 11 days ago	 3149c7a 
Commits on Jul 1, 2015	
 ページングの指示追加 azgz authored 12 days ago	 3a7876d 
Commits on Jun 30, 2015	
 sql取得・表示処理を実装 azgz authored 13 days ago	 7265295 
 phpの実装指示を追加 azgz authored 13 days ago	 ab40b2b 
 Topページhtmlをコミット azgz authored 13 days ago	 4335f6f 



## リビジョン管理システムを使える技術者はイケテいる

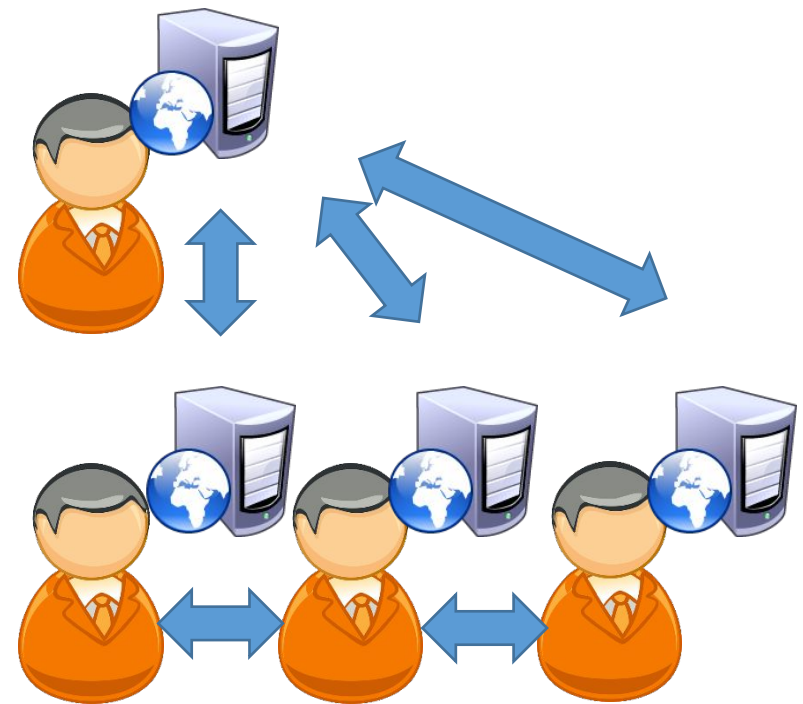
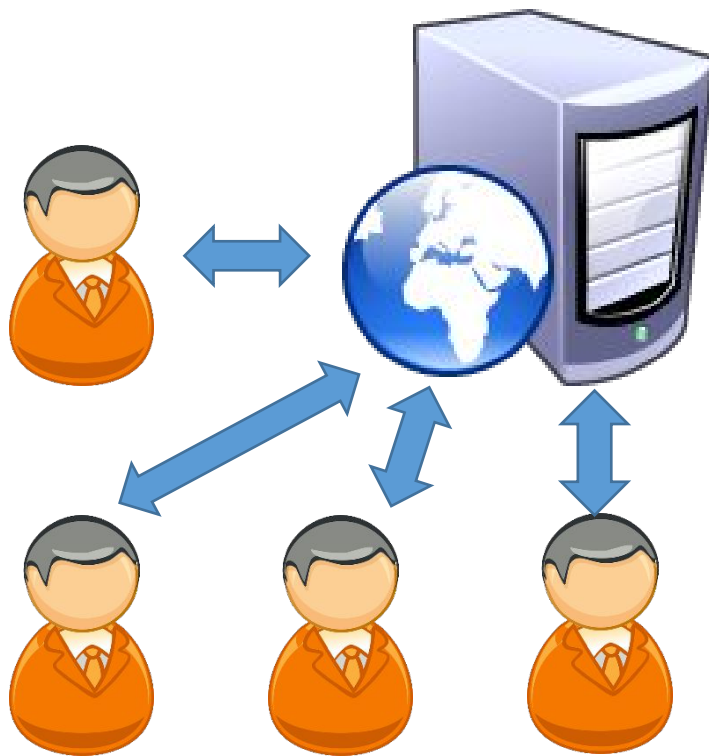
(キマイラ飼育記 2009.12.1)

“ある程度の経験を積んだ技術者／プログラマであるかどうかを判断したいとき、「リビジョン管理システムを普通に使えるかどうか？」という基準はけっこう有効な気がした。”

# Git とは

- Git はバージョン管理システムの一つ
- Git は**分散型**のバージョン管理システム
- Linux のソースコードを管理するために、  
リーナス・トーバルズによって開発された
- 似たバージョン管理システムに、Mercurial (hg)がある
- 少し前までは、集中型バージョン管理(CVS, Subversion)

# 集中型と分散型の違い(イメージ)

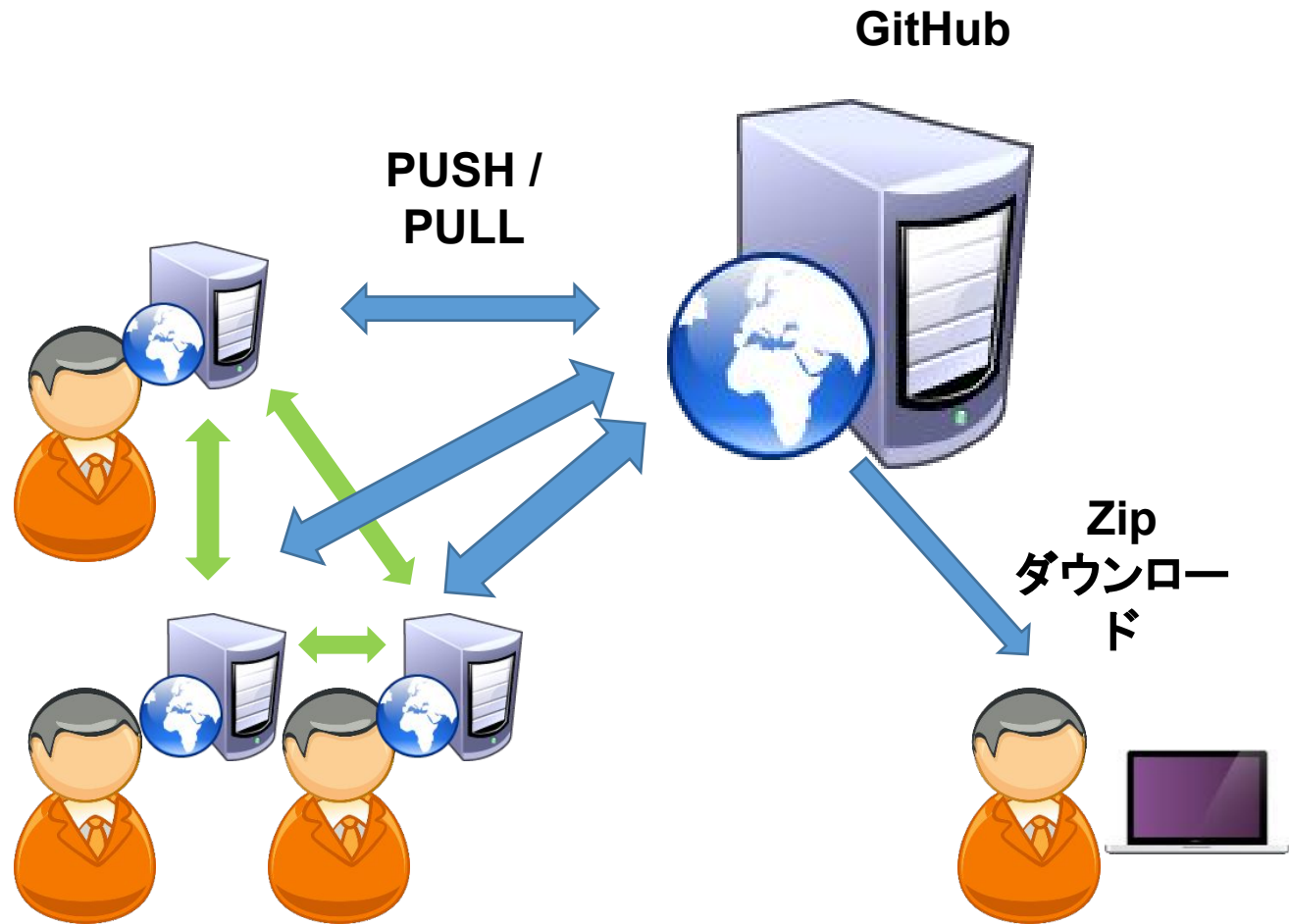


# 良く聞く GitHubって

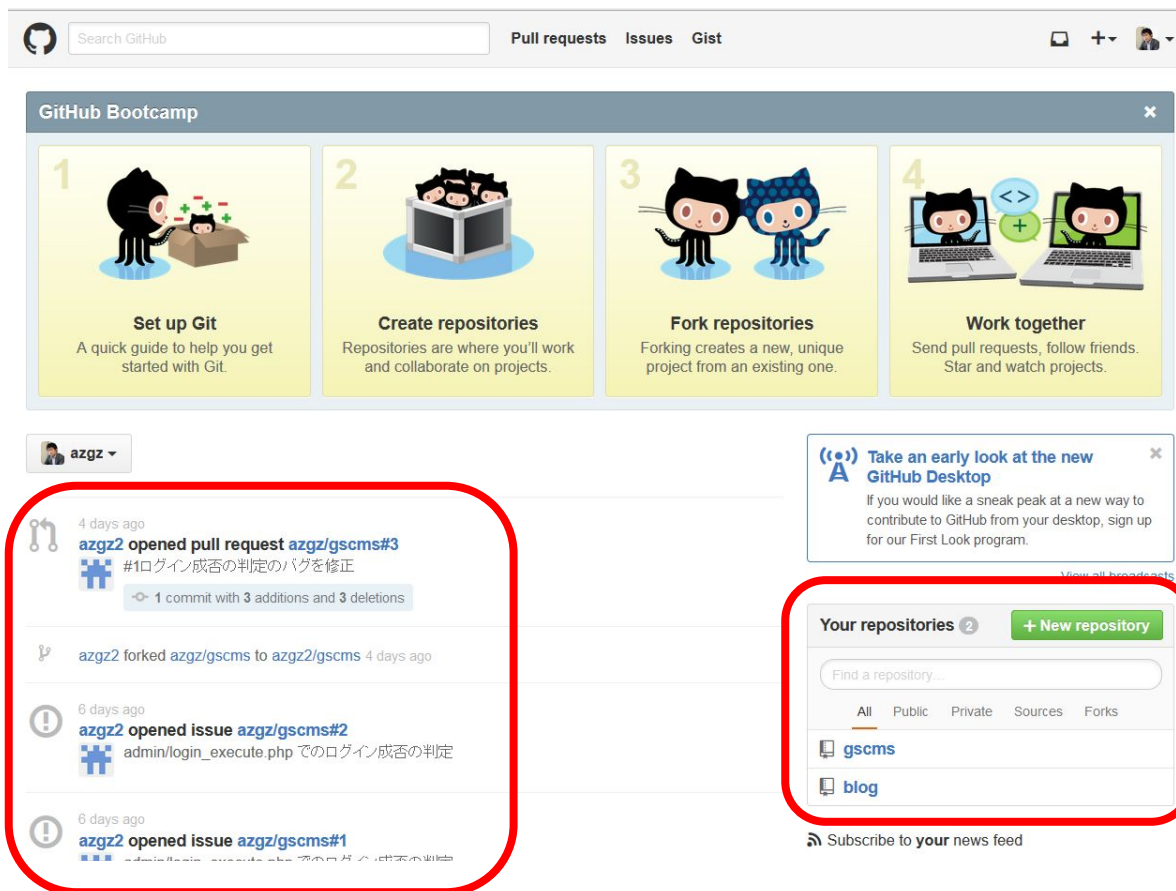
- GitHub は Git のホスティングサイト(Webサービス)
- Ruby on Rails と Erlang で実装(されてたはず)
- バージョン管理情報がブラウザ上で確認できる
- バグ報告など、SNS的な機能もある
- **Fork and Pull Request の仕組み**
- 競合は Bitbucket (非公開リポジトリが無料で作れる)



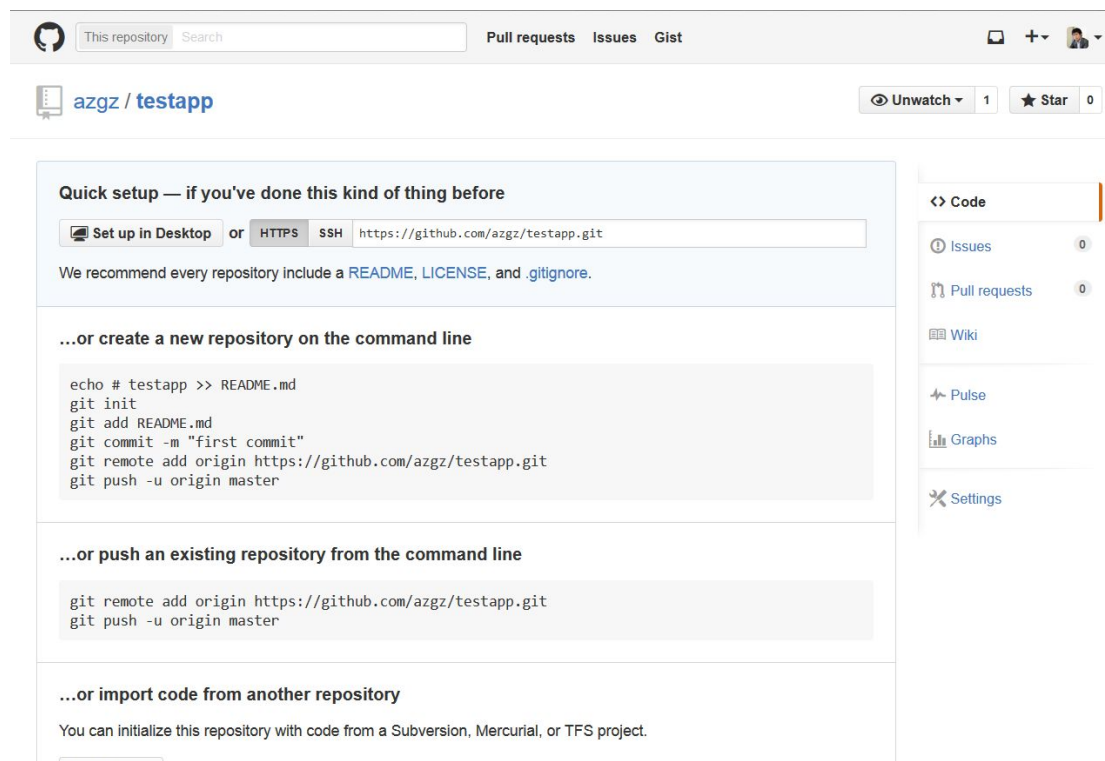
# GitHubのイメージ



# GitHub の画面説明



# GitHub の画面説明



何もファイルを管理していない状態

# GitHub の画面説明

<https://github.com/gs-teacher>

The screenshot shows the GitHub profile page for 'gs-teacher'. The page layout includes a header with a search bar and navigation links (Pull requests, Issues, Gist). The main content area is divided into sections: a profile card on the left, a 'Public contributions' calendar and summary in the center, and a 'Contribution activity' section at the bottom. Red boxes highlight specific elements: the 'Follow' button in the top right, the 'Public contributions' section (including the calendar and summary), the 'Followers', 'Starred', and 'Following' counts in the bottom left, and the 'Contribution activity' section at the bottom.

Search GitHub

Pull requests Issues Gist

+ -

Follow

Contributions Repositories Public activity

Public contributions

Aug Sep Oct Nov Dec Jan Feb Mar Apr May Jun Jul Aug

Summary of pull requests, issues opened, and commits. [Learn how we count contributions.](#) Less More

Contributions in the last year	Longest streak	Current streak
0 total Aug 23, 2014 – Aug 23, 2015	0 days No recent contributions	0 days No recent contributions

Contribution activity

Period: 1 week

gs-teacher has no activity during this period.

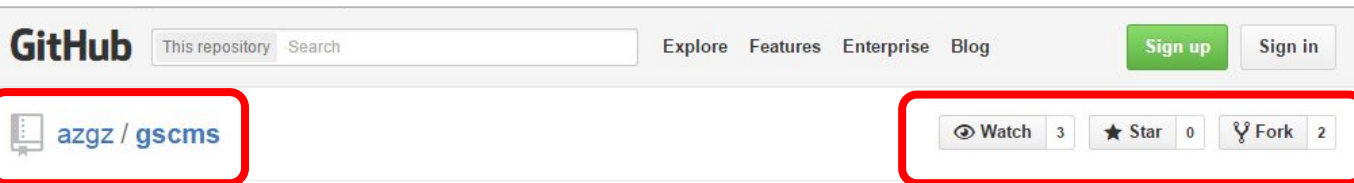
フォロー

リポジトリ  
一覧

フォロー/  
フォロワー

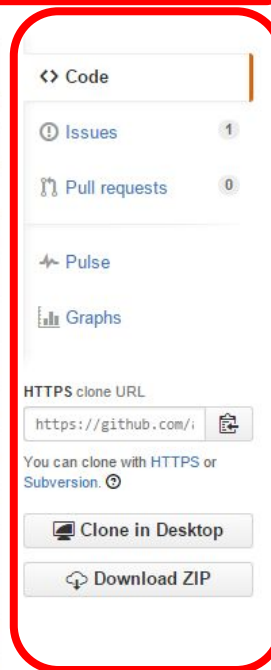
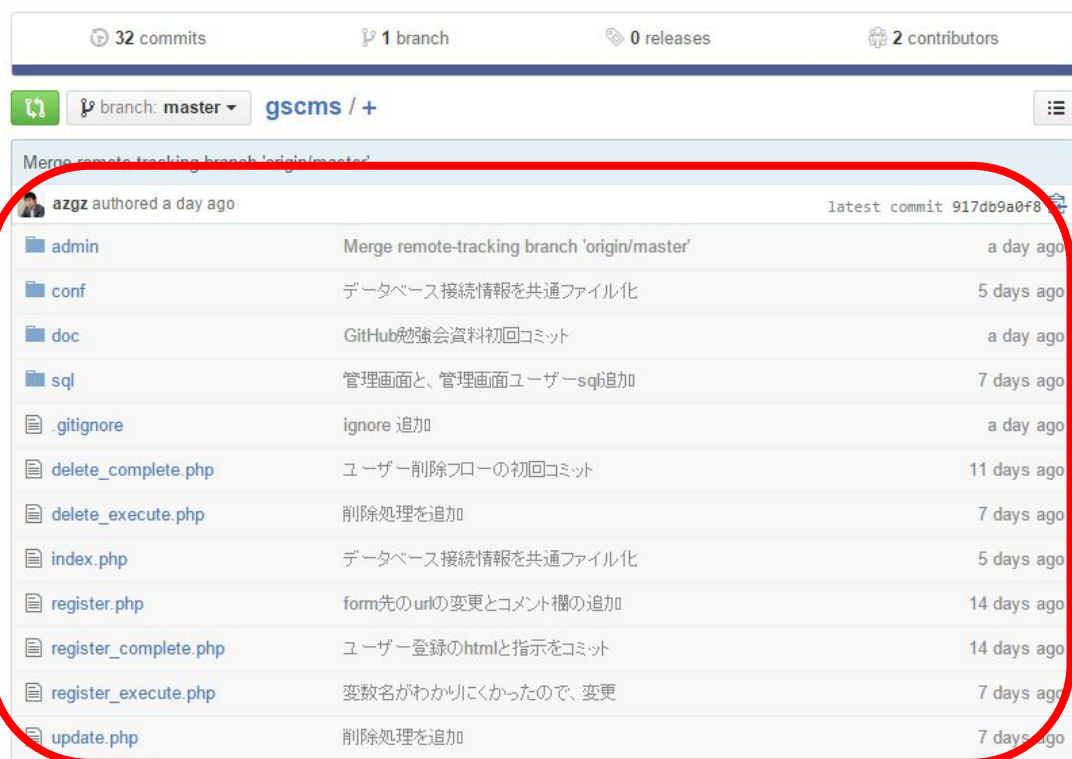
# GitHub の画面説明 (リポジトリ)

リポジトリ名

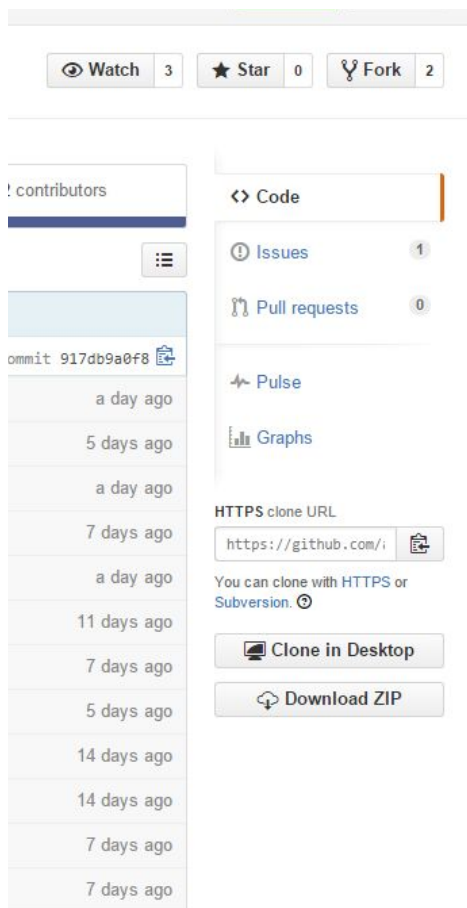


SNS

ソース



# GitHub の画面説明 (SNS機能)



- **Watch**
  - 登録したリポジトリの通知 (イベント)がNotifications画面に表示される。
  - 登録したメールアドレスにメール通知がくる。
- **Star**
  - お気に入り
- **Fork**
  - 自分のアカウント内に既存のリポジトリを複製する
- **Issues**
  - 問題提起や、ディスカッションの場
  - Markdown(マークダウン)形式で記述可能
- **Pull requests**
  - Fork 元のオリジナルなりリポジトリの管理者への通知
- **HTTPS clone URL**
  - 各種クライアントで利用する URL
- **Download ZIP**
  - Zip形式でリポジトリのファイルをダウンロード

# GitHub を使った流れ(1)



1. GitHub アカウント作成

2. オリジナルなりポジトリを作成



3. オリジナルなりポジトリをローカル環境(mac/PC)に clone

4. clone したローカル環境で開発

5. ローカル環境で更新したファイルを、ローカル環境のgit にコミット



6. ローカル環境から、GitHub の オリジナルなりポジトリに Push

# GitHub を使った流れ(2)



1. GitHub アカウント作成

2. どこかのリポジトリを、自分のアカウントに fork



3. fork したリポジトリをローカル環境(mac/PC)に clone

4. clone したローカル環境で開発

5. ローカル環境で更新したファイルを、ローカル環境のgit にコミット



6. ローカル環境から、GitHub の fork したリポジトリに Push

7. fork したリポジトリから、元のリポジトリに Pull Request




# Git にコミットしてみよう

1. GitHub 上に kadai リポジトリを作成
2. URL をコピー
3. SourceTree で「新規/クローンを作成する」
4. 「元のパス/URL」に、2のURLをペースト、「保存先のパス」に DocumentRoot(XAMPPのhtdocs より下)を指定
5. 該当フォルダを開いて、「6」という名前のディレクトリを作成
6. その後、「6」ディレクトリの中にindex.php を作成(コピペ可)

# Git にコミットしてみよう

GitHub 上に kadai リポジトリを作成

Owner:  azgz / Repository name:

Great repository names are short and memorable. Need inspiration? How about [equanimous-octo-prune](#).

Description (optional)

☒ **Public**  
Anyone can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** | Add a license: **None** ⓘ

[Create repository](#)

[Terms](#) [Privacy](#) [Security](#) [Contact](#) [Help](#)

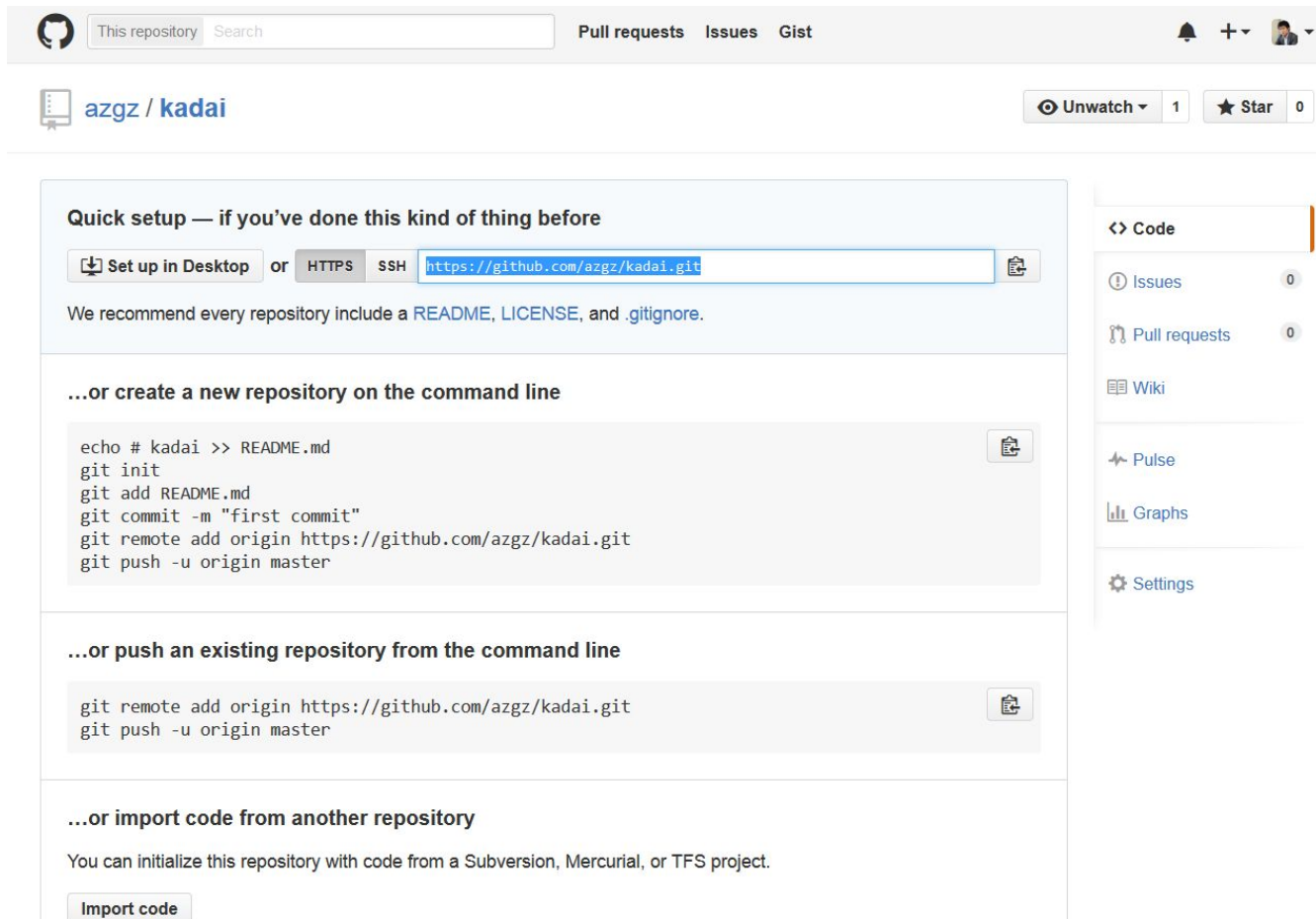


[Status](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)



# Git にコミットしてみよう

URL をコピー



The screenshot shows the GitHub interface for the repository 'azgz / kadai'. The top navigation bar includes the GitHub logo, a search bar, and links for 'Pull requests', 'Issues', and 'Gist'. The repository name 'azgz / kadai' is displayed, along with 'Unwatch' and 'Star' buttons. The main content area is titled 'Quick setup — if you've done this kind of thing before' and provides instructions for cloning the repository. It includes a 'Set up in Desktop' button, a selection of 'HTTPS' and 'SSH' protocols, and a text input field containing the HTTPS URL 'https://github.com/azgz/kadai.git'. Below this, it recommends including a README, LICENSE, and .gitignore file. The section also provides a command-line setup for creating a new repository, including commands for initializing the repository, adding the README file, committing the changes, adding the remote origin, and pushing the code to the origin master branch. A sidebar on the right contains links to 'Code', 'Issues', 'Pull requests', 'Wiki', 'Pulse', 'Graphs', and 'Settings'.

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH <https://github.com/azgz/kadai.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo # kadai >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/azgz/kadai.git
git push -u origin master
```

...or push an existing repository from the command line

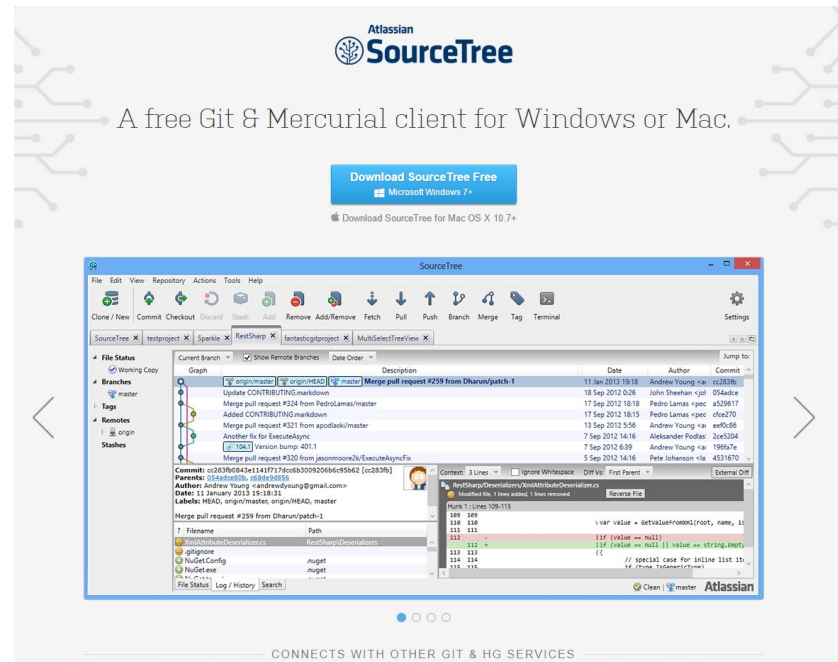
```
git remote add origin https://github.com/azgz/kadai.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

# SourceTree の紹介



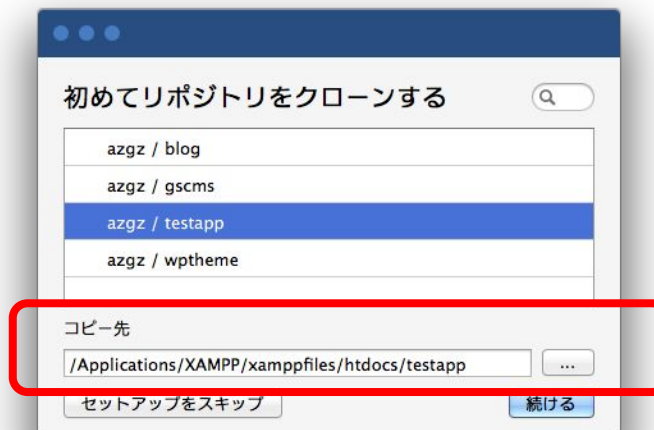
Git クライアント

TOWER<sup>2</sup>

universions

SourceTree は有名な Git クライアント  
Git 自体はコマンドラインでの作業を想定している(CUI)が、  
マウスでわかりやすく操作することを可能(GUI)にしたソフト

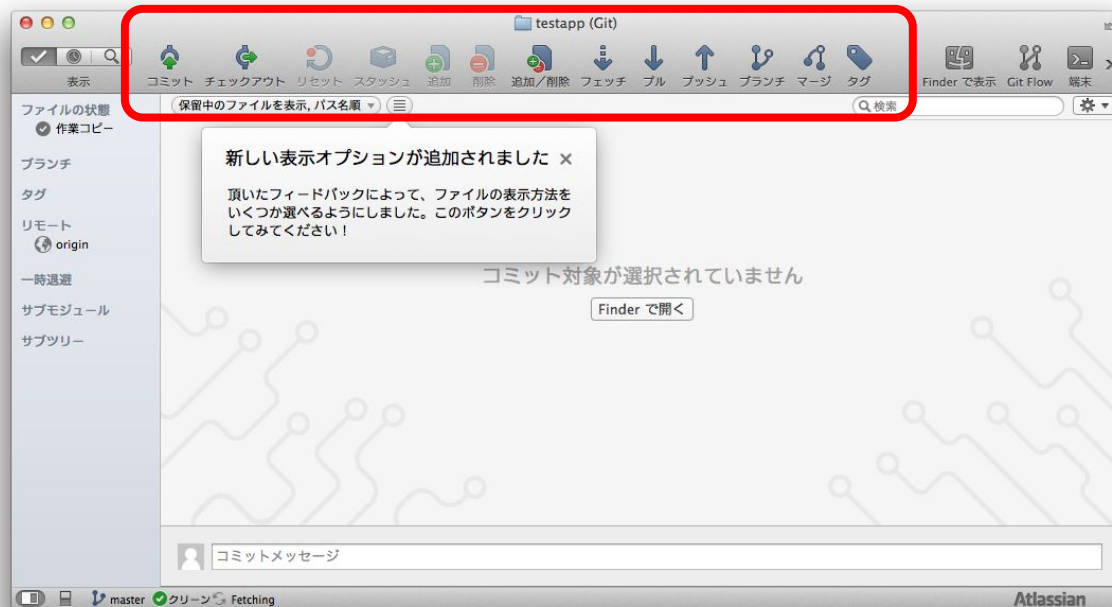
# SourceTree の紹介



最初に起動時は、アカウントを追加するか聞かれるので、追加する。

次に、リポジトリがあれば、それをクローン(ローカル)に持ってくる  
(※コピー先を開発環境に持ってくると、作業がやりやすい)

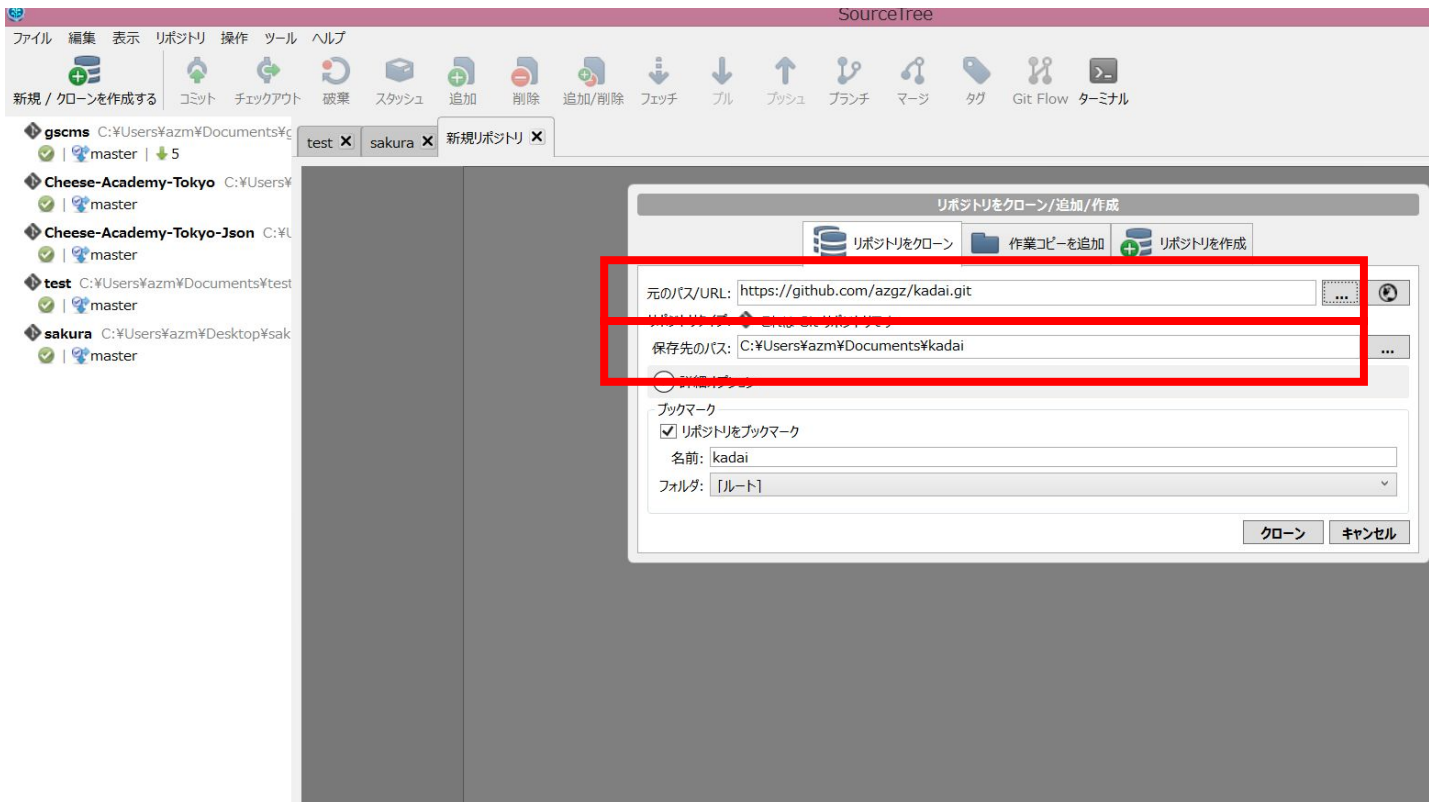
# SourceTree の紹介



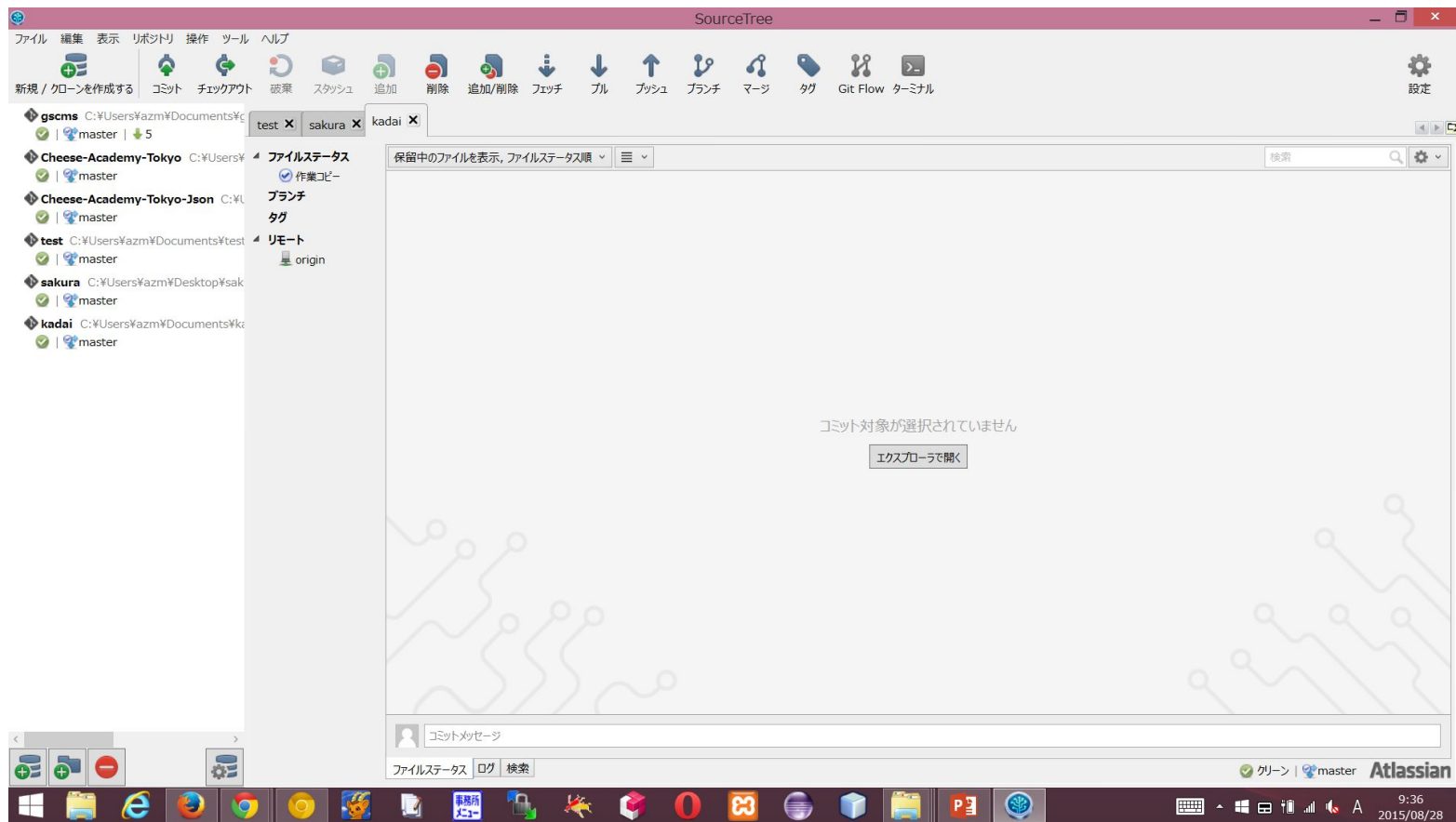
- コミット
  - 更新したファイルと履歴をローカルに登録
- プル
  - 変更の取得とマージ
- プッシュ
  - GitHubへの登録

# Git にコミットしてみよう

URL をペースト、  
保存先のパスに、**htdocs** 以下を指定

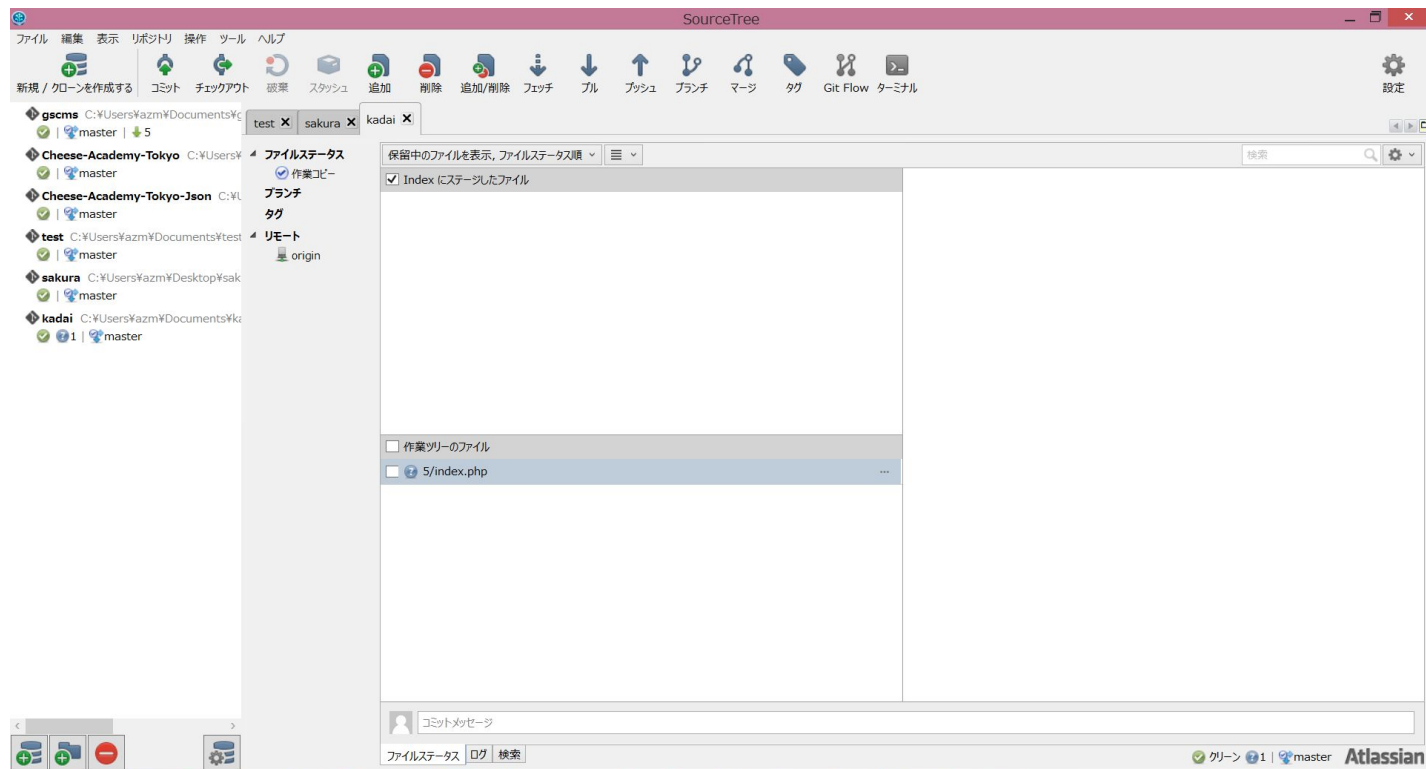


# Git にコミットしてみよう

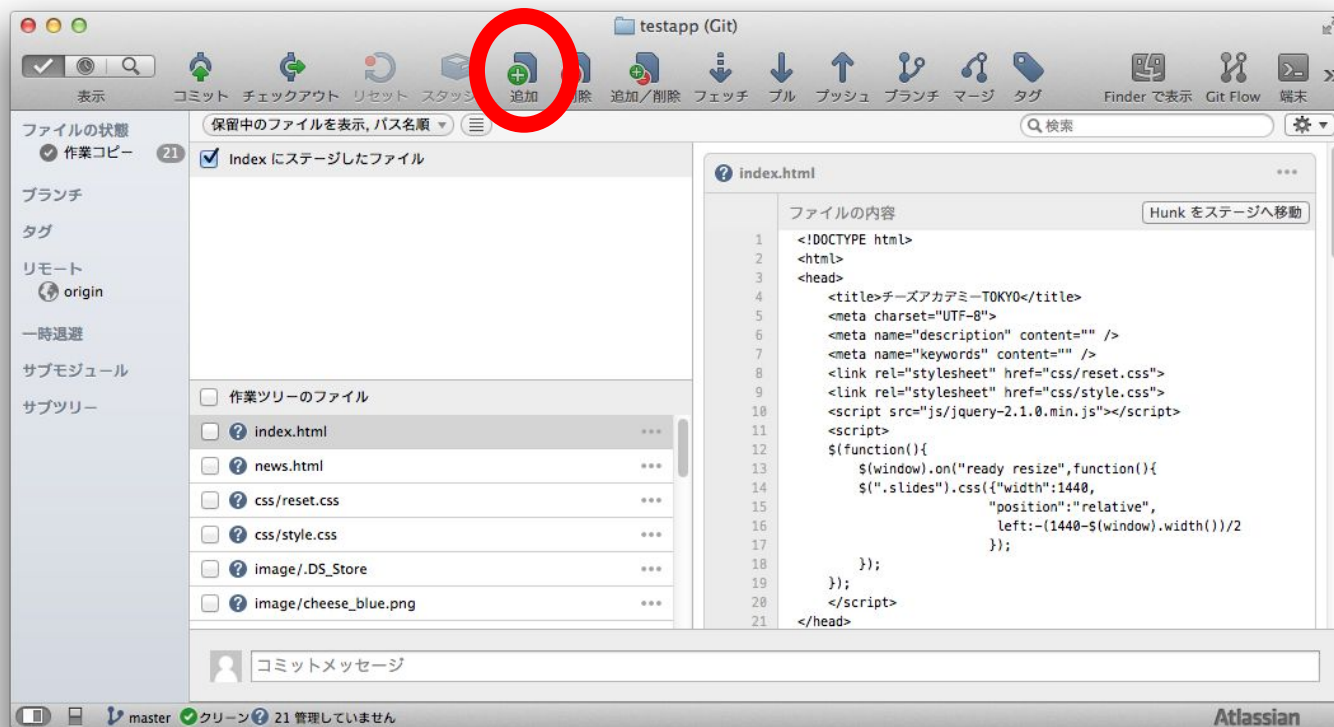




# Git にコミットしてみよう



# Git にコミットしてみよう

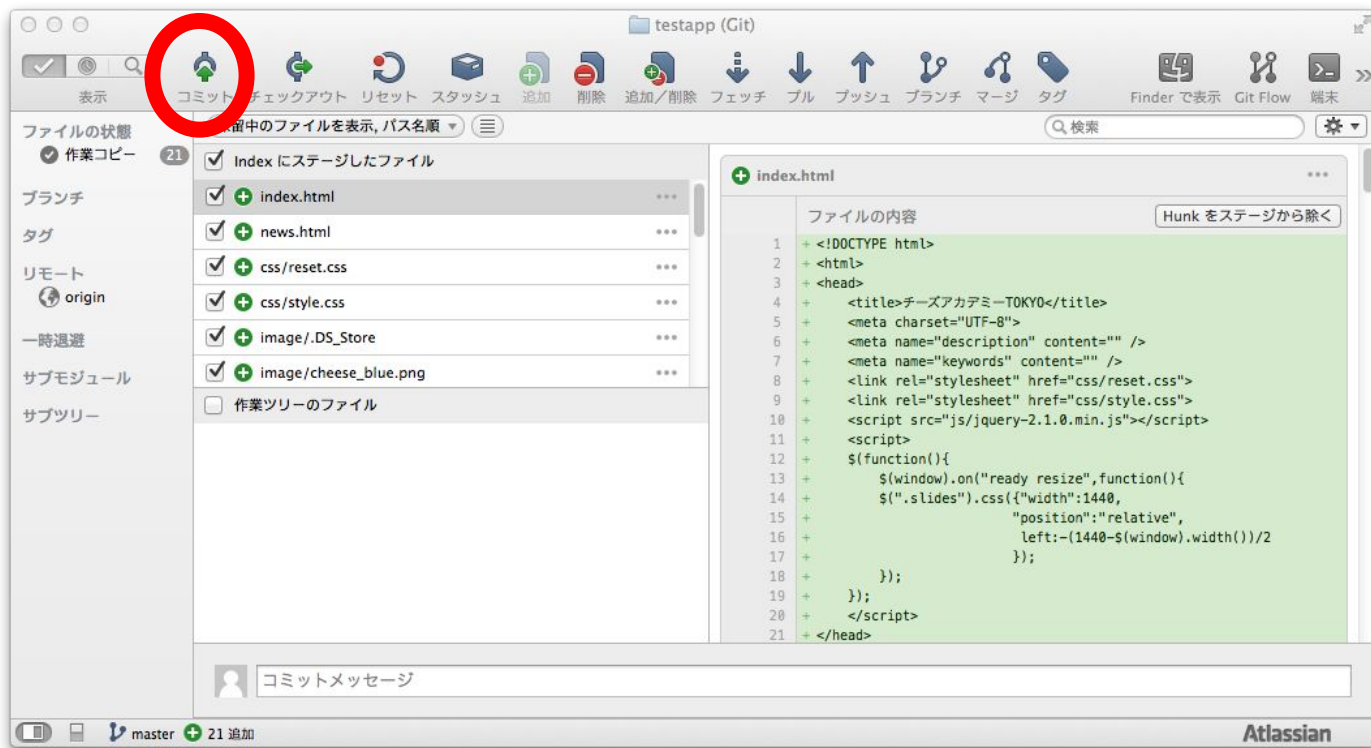


ファイルを編集したり、追加したりすると、「作業ツリーのファイル」へ 編集・追加されたファイルの一覧がでてくる。ファイルを選択して、「追加」

?マーク: 今回初めて追加されたファイル

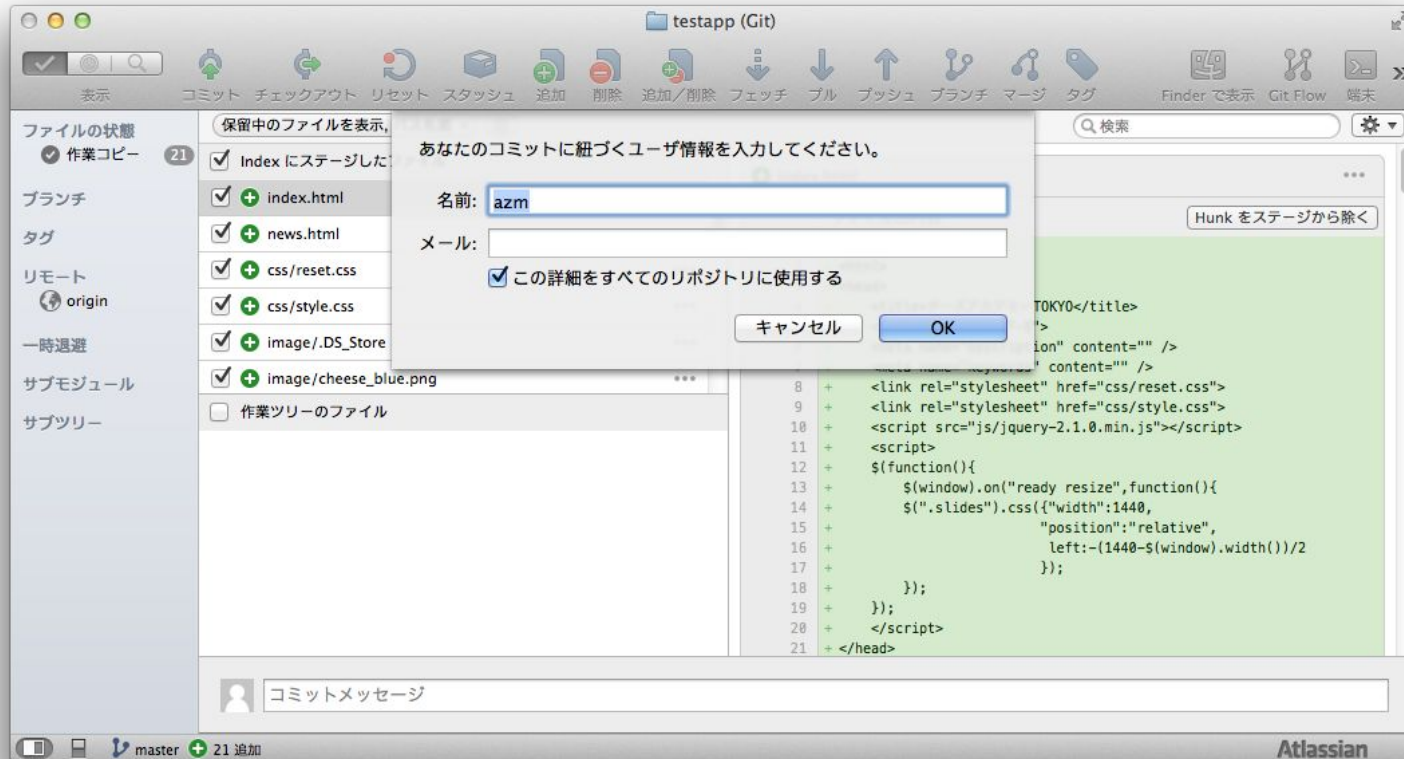
...マーク: 編集されたファイル

# Git にコミットしてみよう



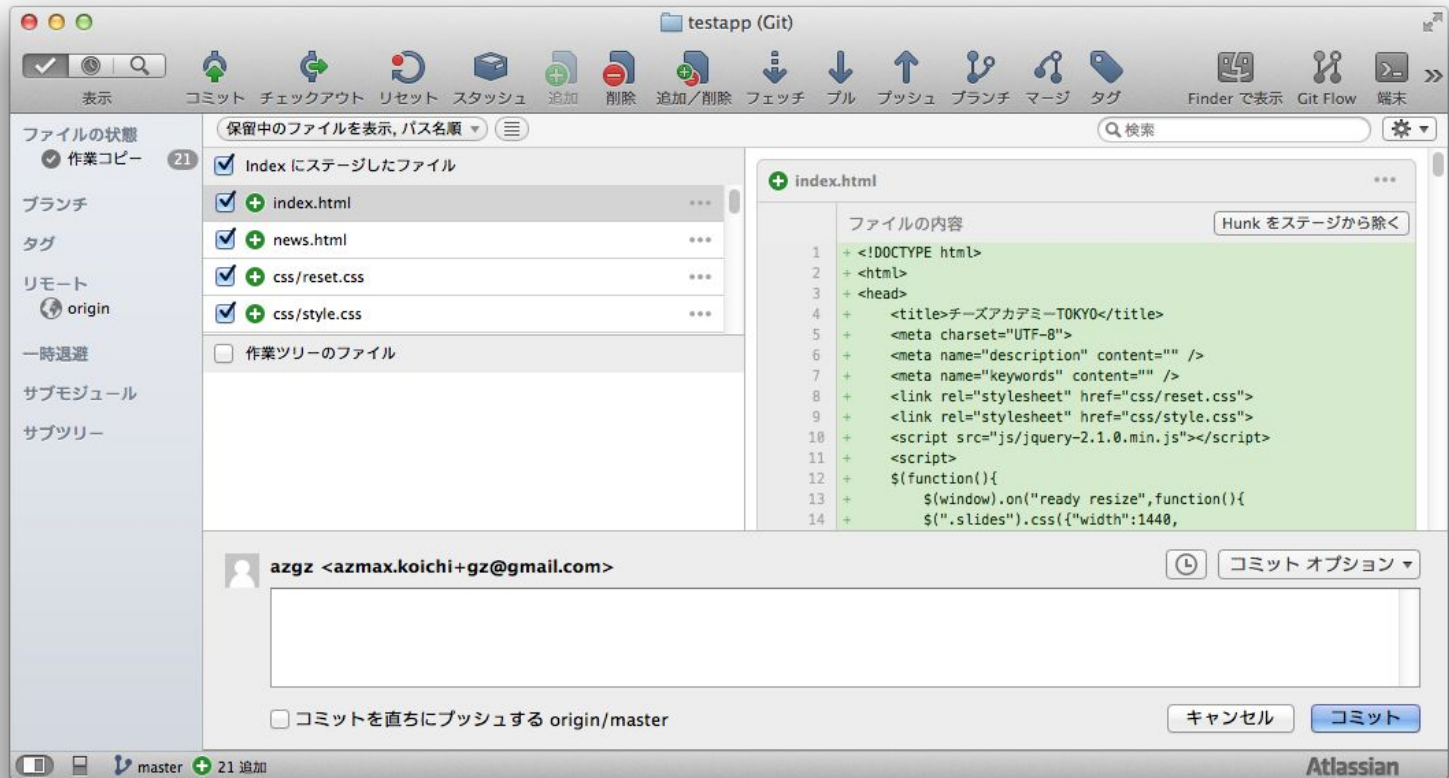
コミットしたいファイルを選択したあと、「コミット」  
※作業単位での選択がオススメ

# Git にコミットしてみよう



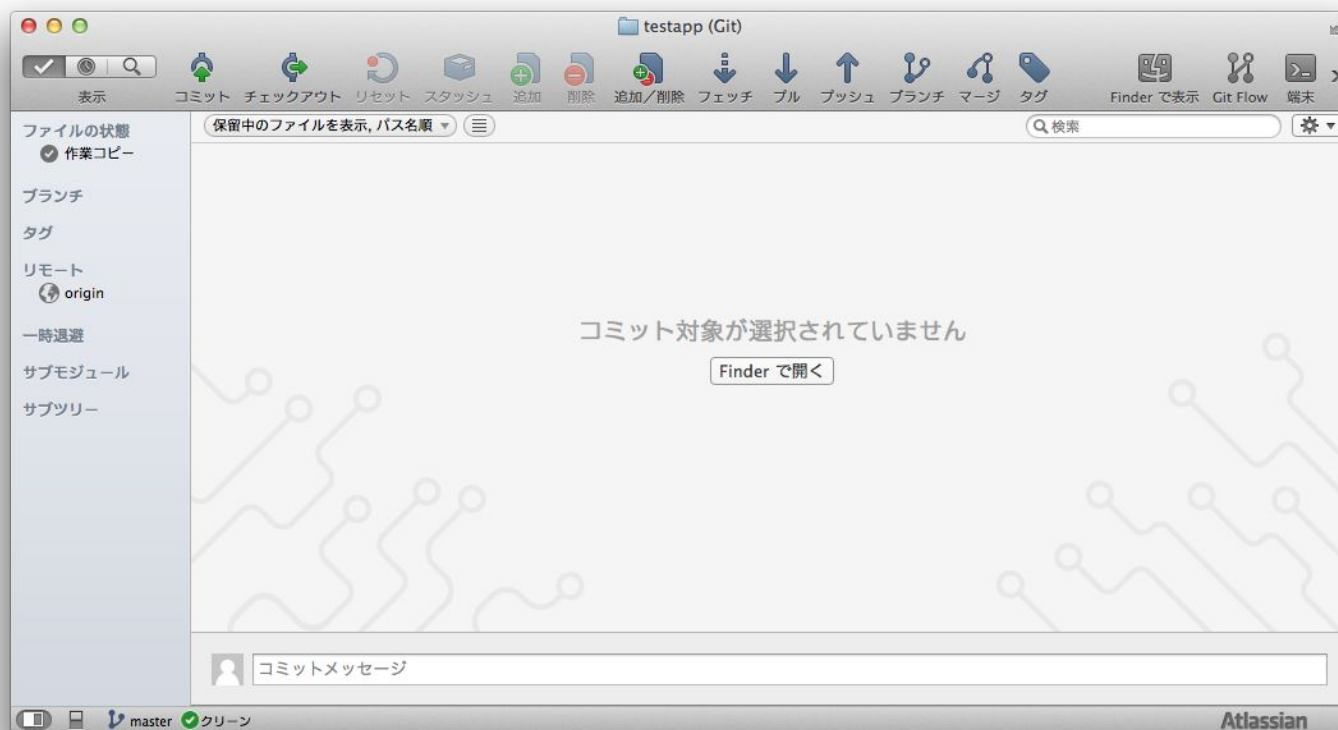
名前とメールアドレスを登録  
※公開されるので、注意

# Git にコミットしてみよう



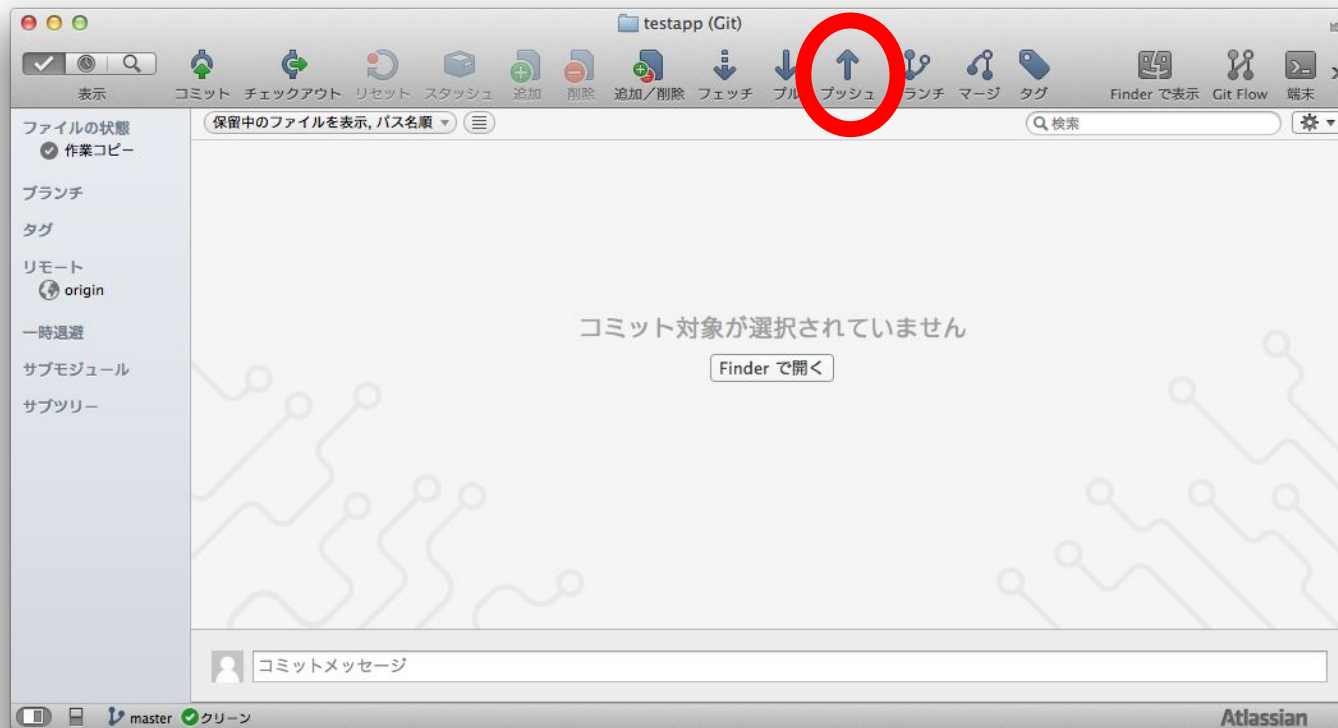
コミットメッセージを記入し、「コミット」

# Git にコミットしてみよう



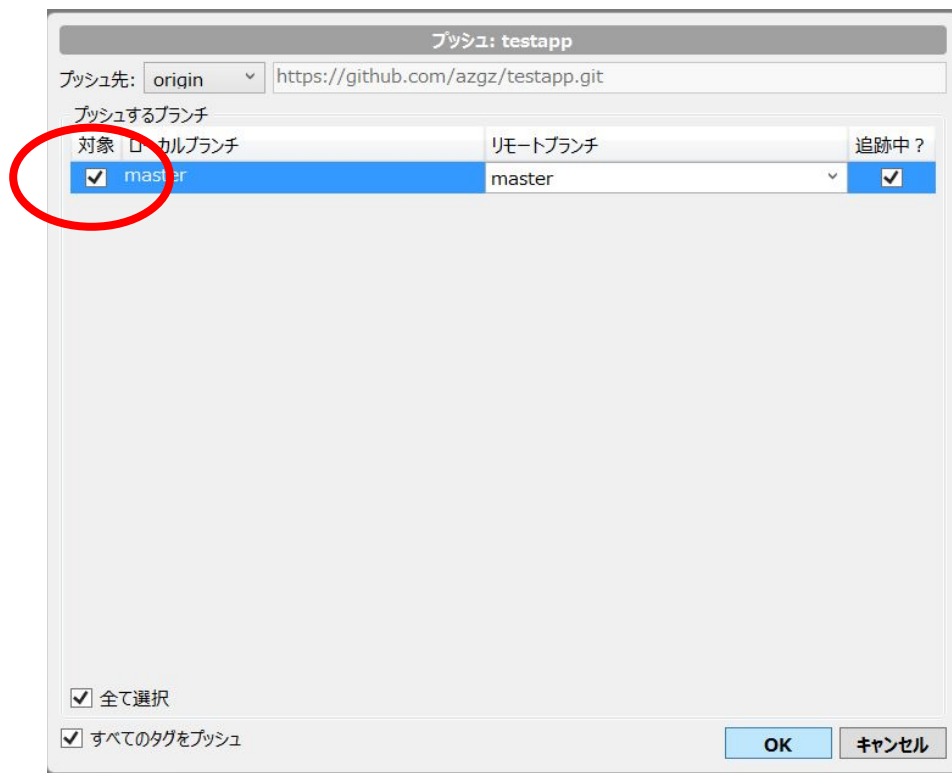
コミット後、作業ツリーにもindex上にもファイルがなくなれば、コミット成功

# GitHub にプッシュしてみよう



プッシュすると、GitHub 上にコミットしたファイルが履歴とともに反映される

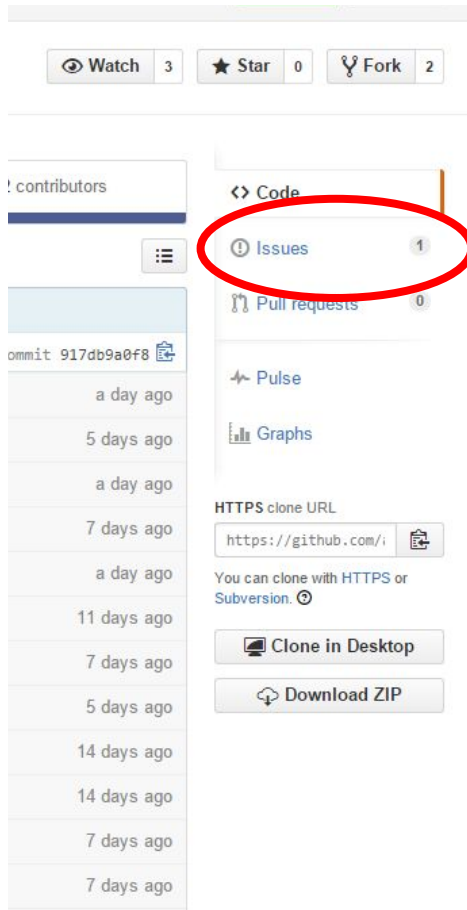
# GitHub にプッシュしてみよう



チェックボックスに、チェックを入れて、OK



# PUSH が終わったら、報告



- **Issues**

- 問題提起や、ディスカッションの場
- Markdown(マークダウン)形式で記述可能
- ここで、提出を報告！

