

サイバー大学IT総合学部

専門応用科目

JavaScriptフレームワークによるWebプログラミング

第3回 Node.js実習1

小園井康志

第3回 学習目標

- GitHubの概要および基本的な使い方を理解し、GitHubからソースコードをダウンロードし、Node.jsのサイトを立ち上げることができる

第3回 授業構成

- 第1章 GitHub概要および実習
- 第2章 nodejsでサイトを作成

JavaScriptフレームワークによるWebプログラミング

第3回Node.js実習1

第1章

GitHub概要および実習

第1章 学習目標

- GitHubの概要を理解し説明できる
- GitHubの基本的な使い方である、以下を理解し実践できる
 - GitHub上でのプログラムの管理、共有
 - ローカル（PC上の）環境との同期
 - GitHub上にリポジトリを作成し、ローカルにclone、編集後GitHubにpush（アップロード）

GitとGitHub

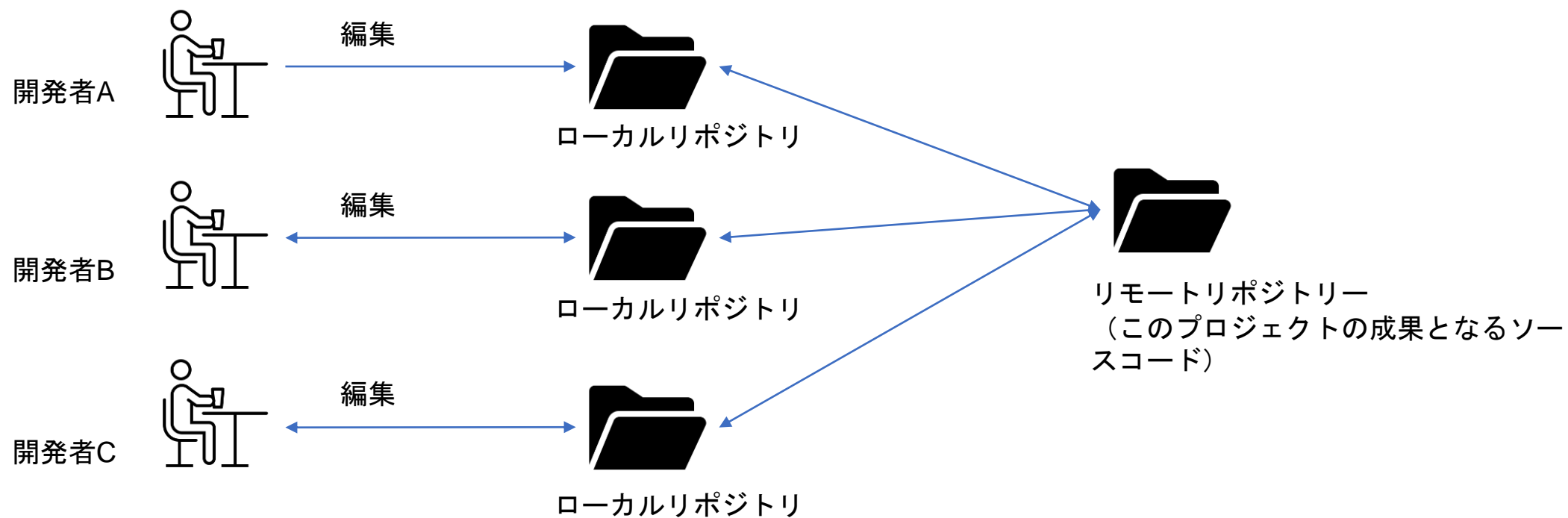
- Git

ソフトウェアの変更履歴管理・バージョンを管理するツール

分散型のツールであり複数の開発者がそれぞれに変更履歴を含む完全なソースコードの複製を作成し変更を加えることができるのが大きな特徴。オープンソースのプロジェクト、Linuxのカーネル開発で使われ始め現在はその他のオープンソースプロジェクト、社内の開発プロジェクトなどでも使われている。

Gitの仕組み

- 各自が各自の端末でソースコードの編集、開発を行う



GitとGitHub

- GitHub

GitHubは、ソースコードのバージョン管理ツール、“Git”を利用した開発者を支援するWebサービスで、バージョン管理とコラボレーションのためのコードホスティングプラットフォーム。

GitHubを使えば、どこにいてもプロジェクトで他の開発者と一緒に作業することができる。多くのオープンソースプログラムがこの仕組みで管理されており、多くの開発者が開発に参加できるようになっている。

この授業ではサンプルプログラムの利用にこのGitHubを利用する。

Gitツールのインストール (Windows)

- <http://git-scm.com/download/win>
- 上記ページから該当プログラムをダウンロード、実行する。

git --everything-is-local

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Download for Windows

Click here to download the latest (2.37.0) 32-bit version of **Git for Windows**. This is the most recent [maintained build](#). It was released **11 days ago**, on 2022-06-27.

Other Git for Windows downloads

Standalone Installer
[32-bit Git for Windows Setup.](#)
[64-bit Git for Windows Setup.](#)

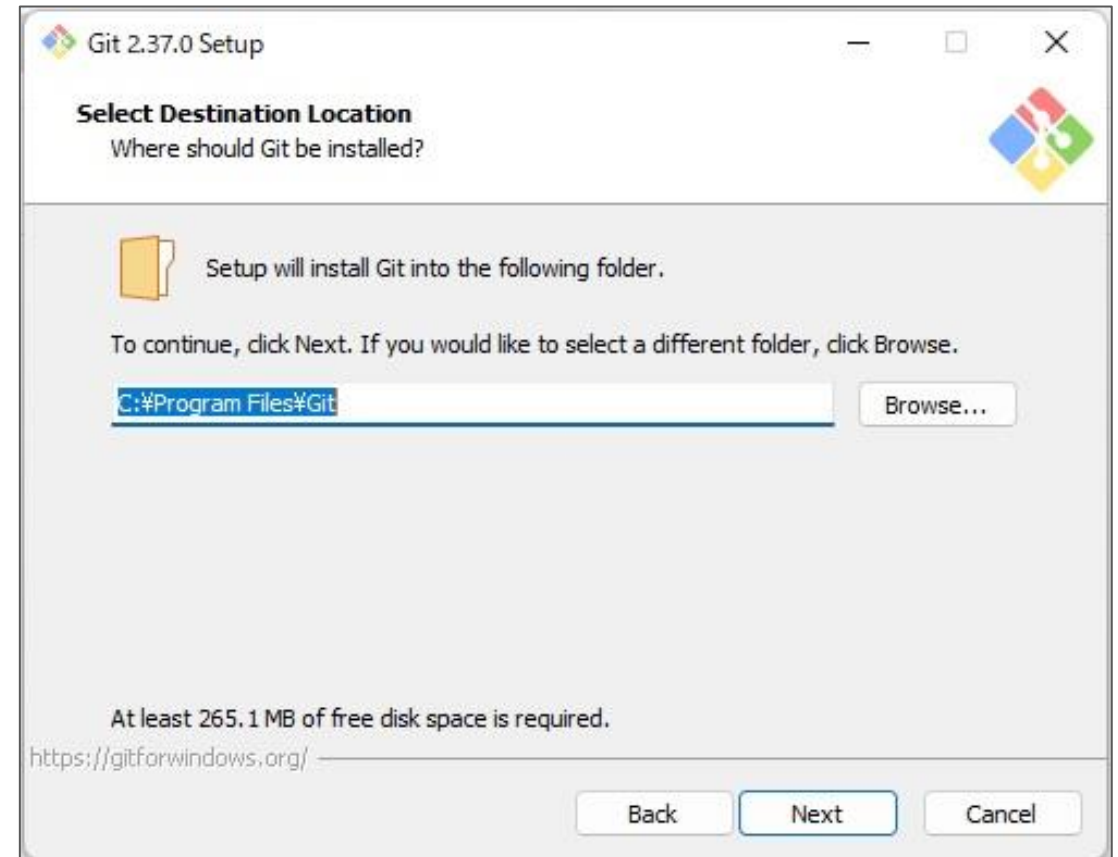
Portable ("thumbdrive edition")
[32-bit Git for Windows Portable.](#)
[64-bit Git for Windows Portable.](#)

Using winget tool

ここをクリックすると
最新版がダウンロード
できる。

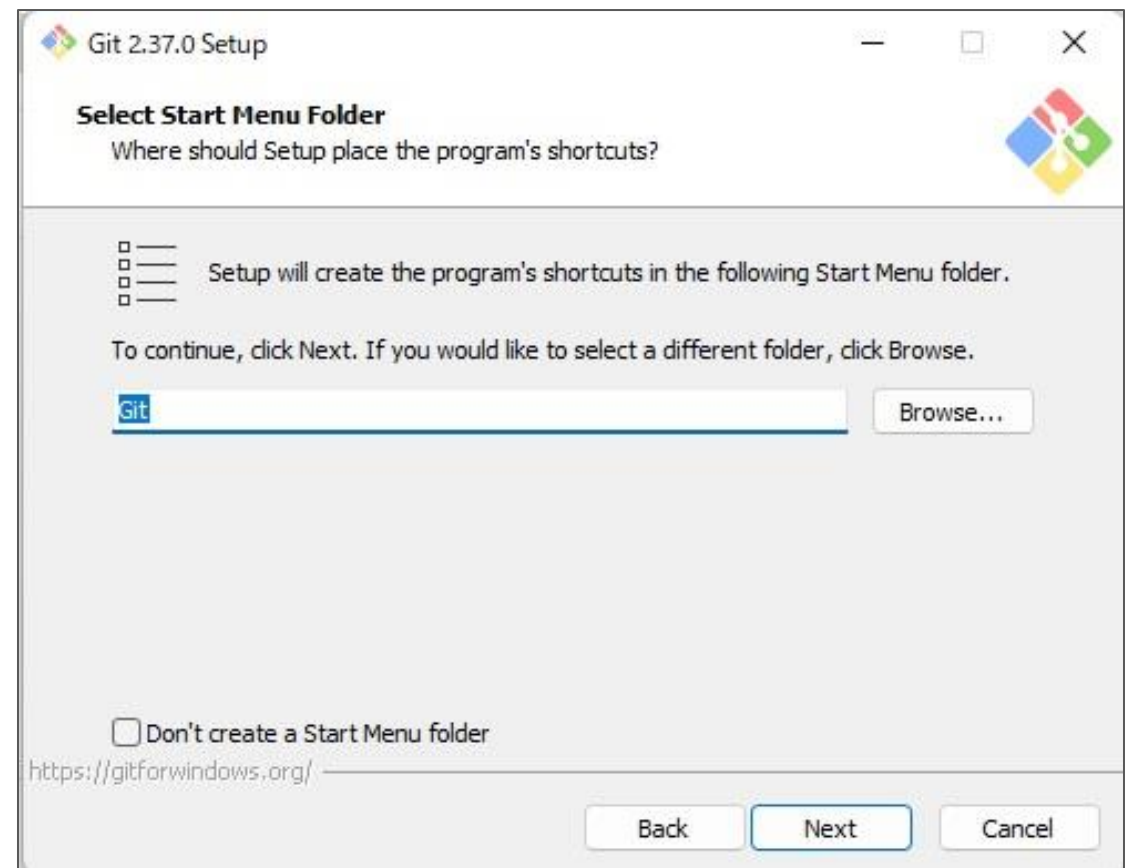
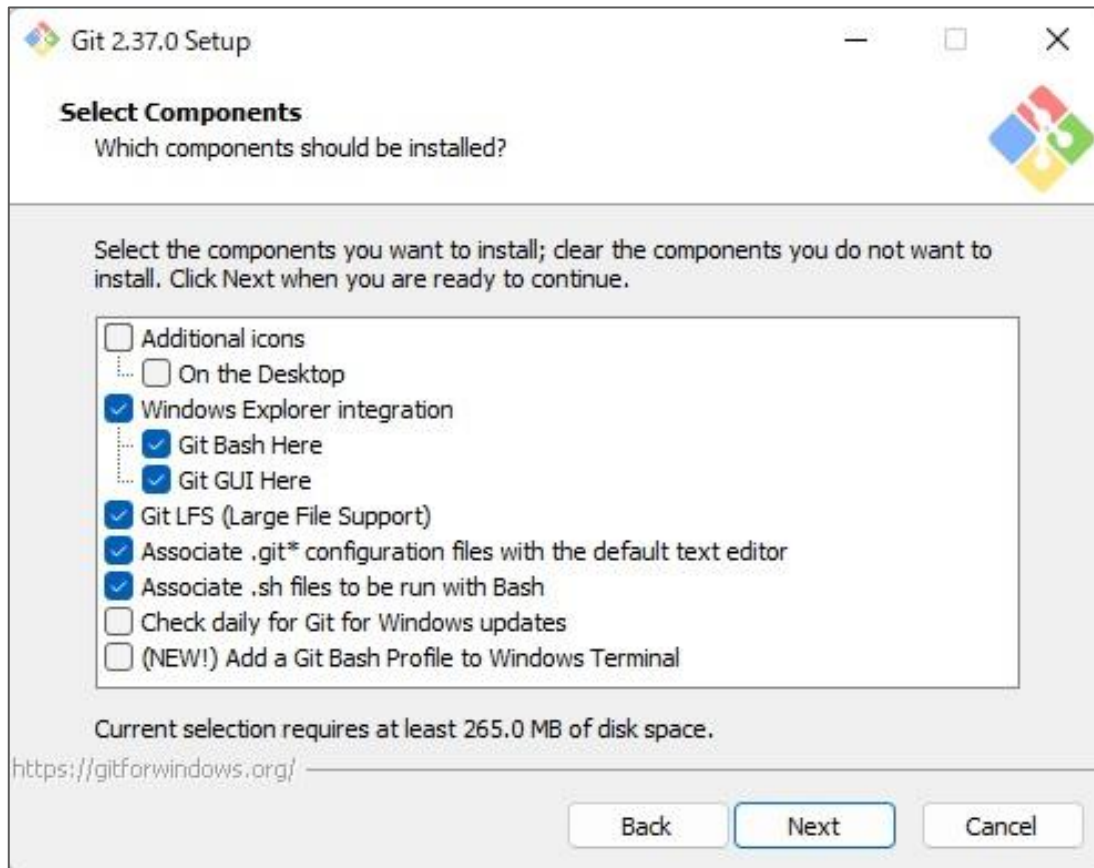
Gitツールのインストール (Windows)

- インストーラーが立ち上がったら“Next”をクリック
- 次の画面から全てデフォルト変更せずに“Next”をクリック



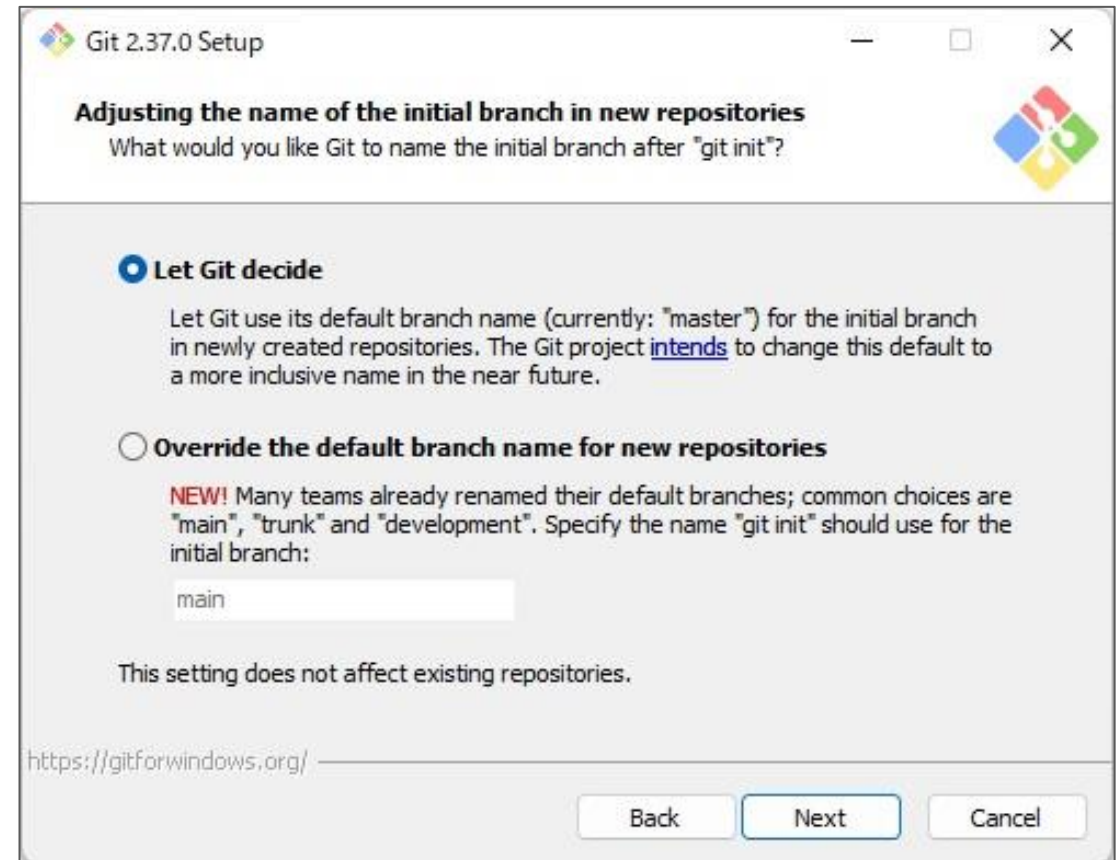
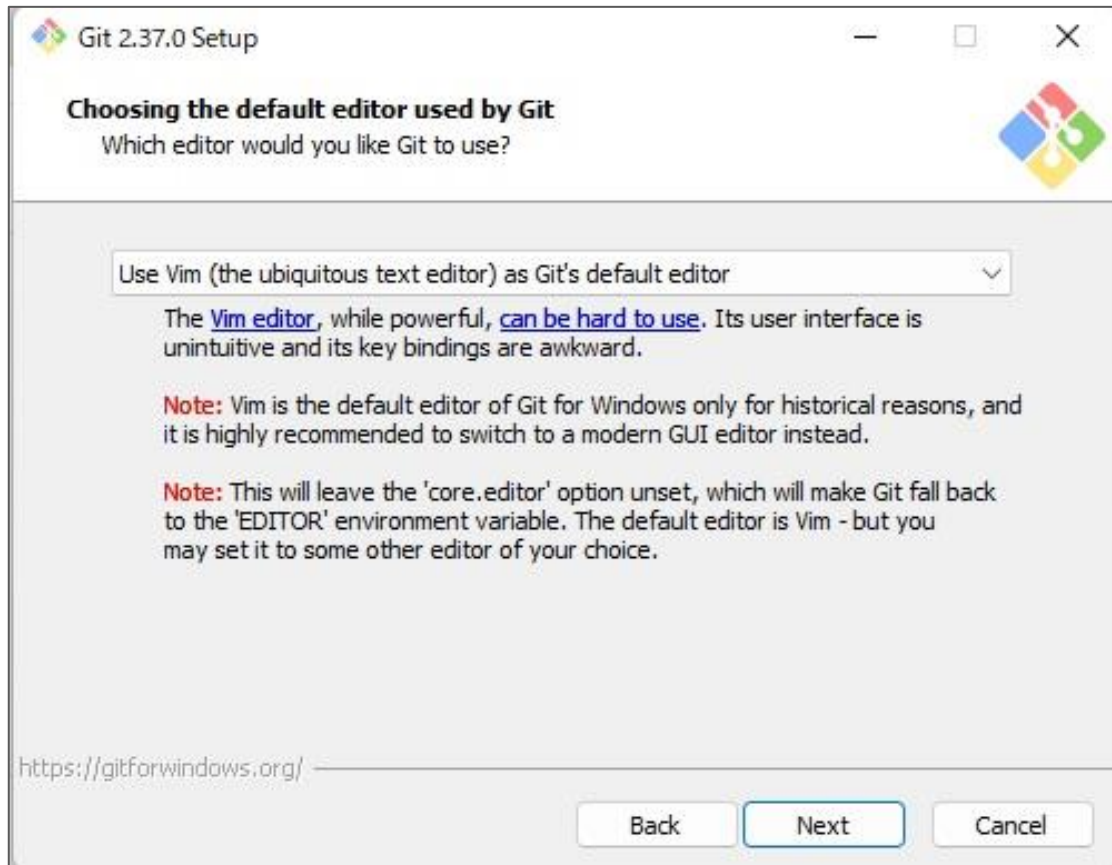
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



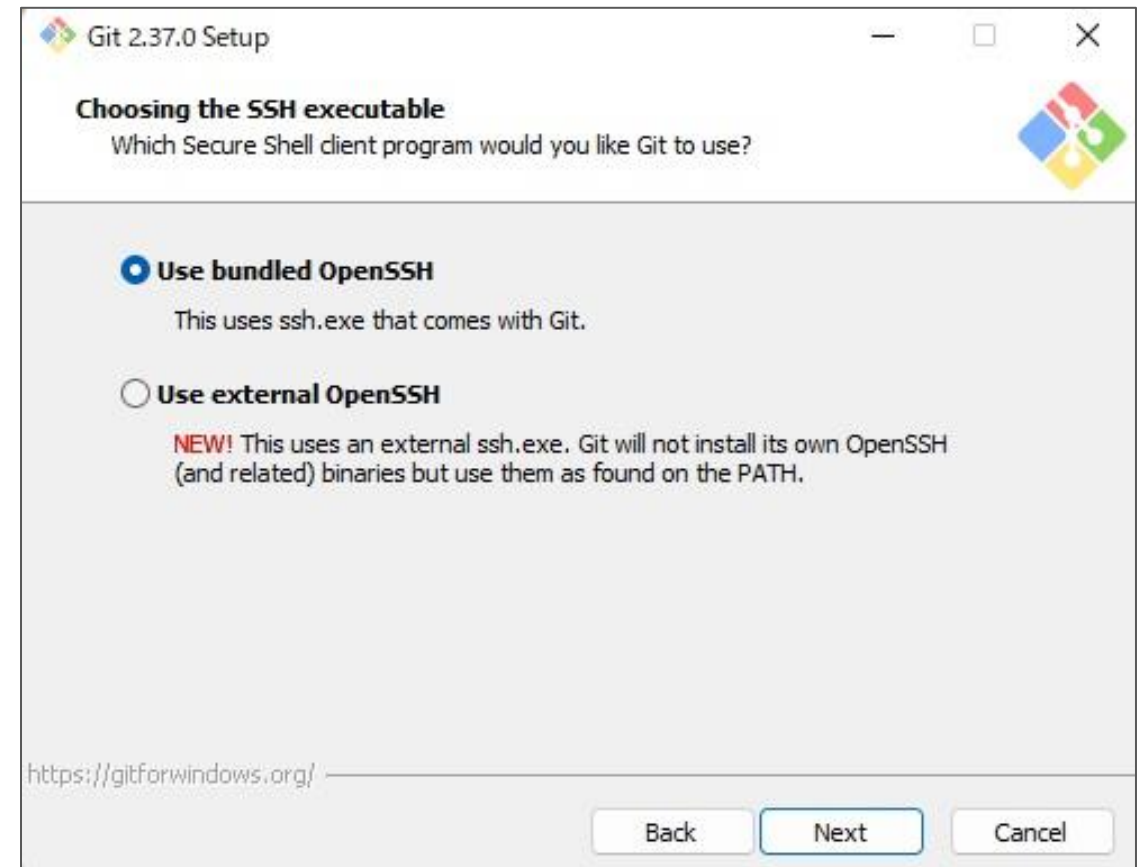
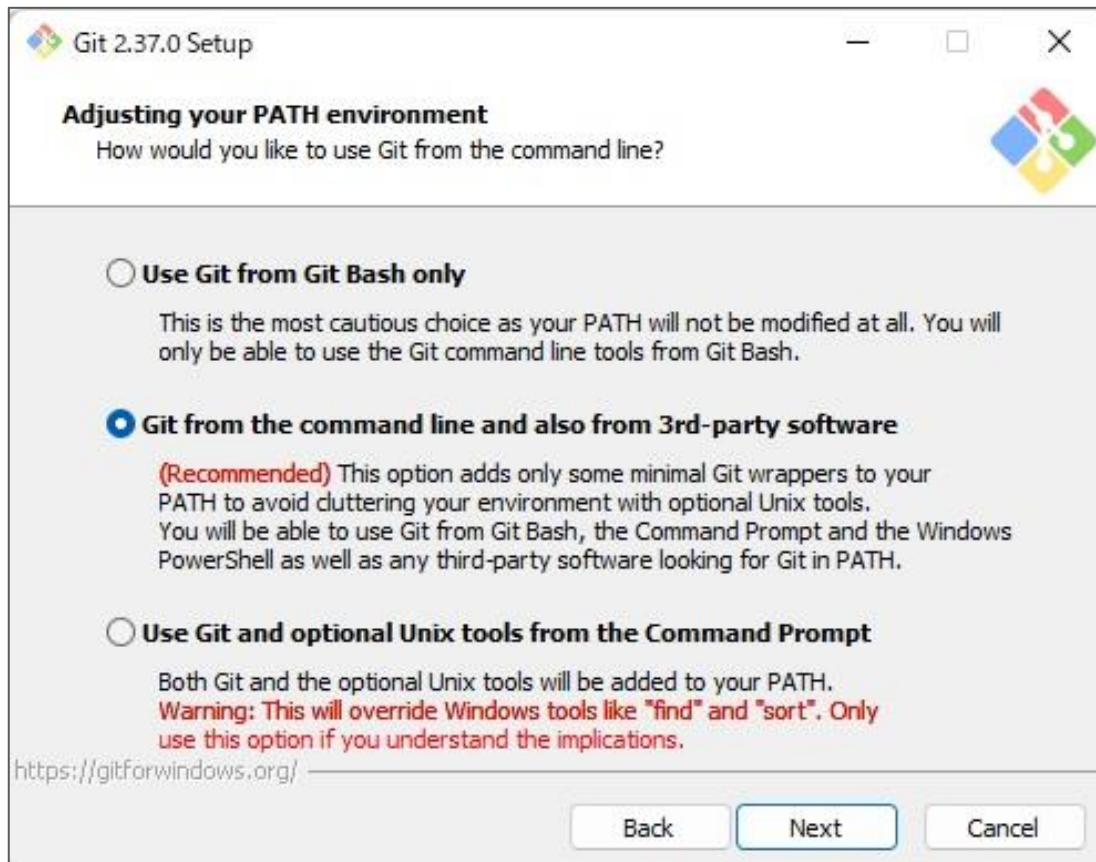
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



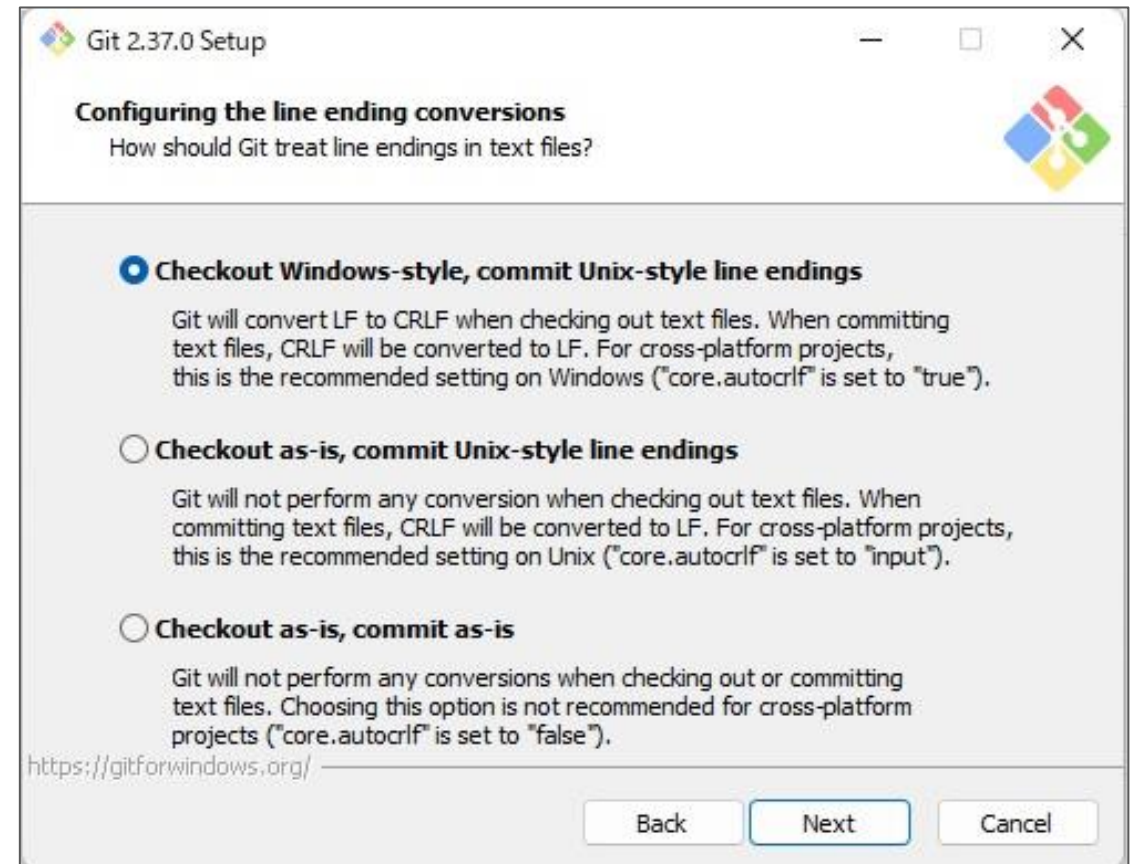
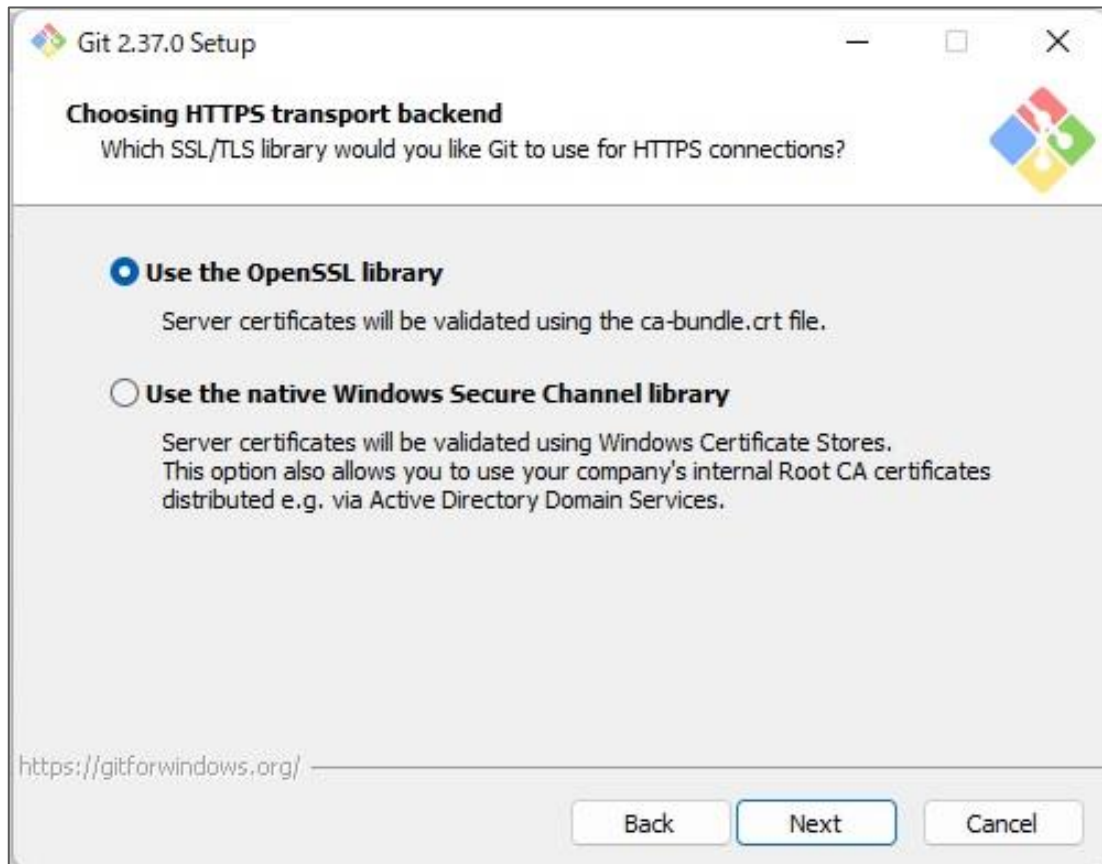
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



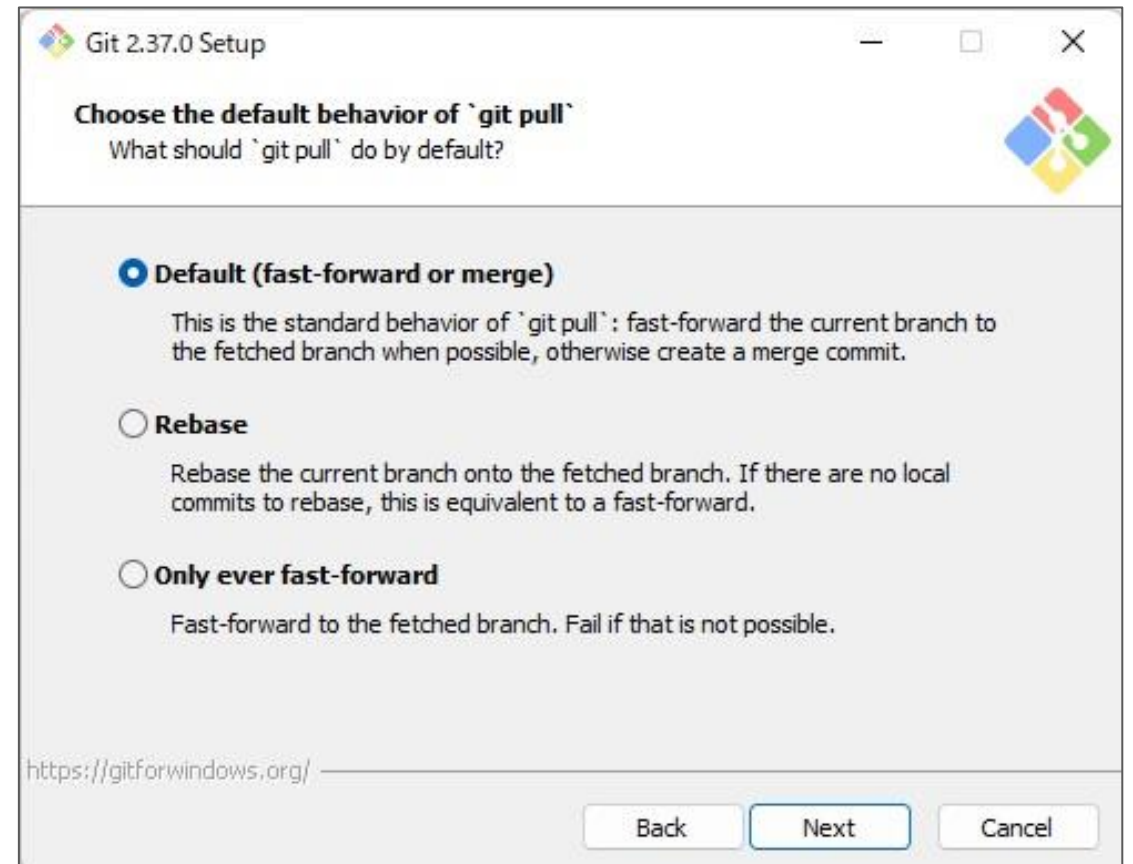
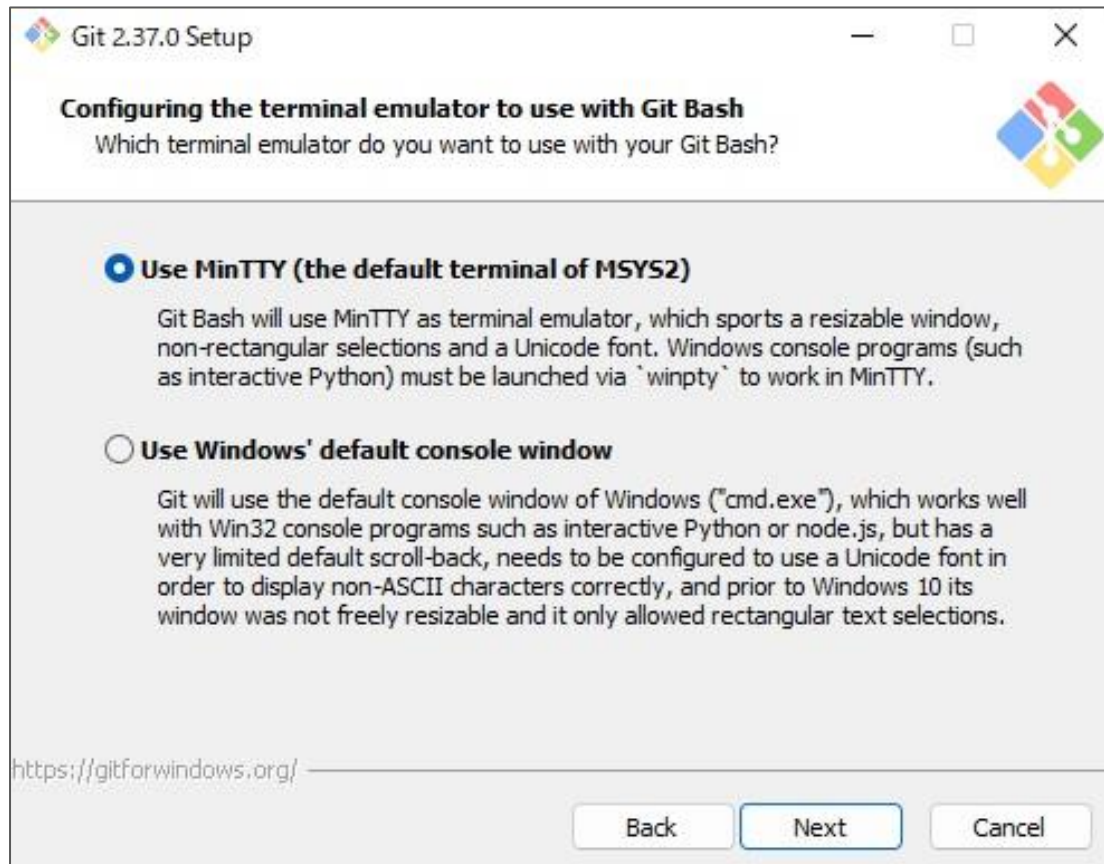
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



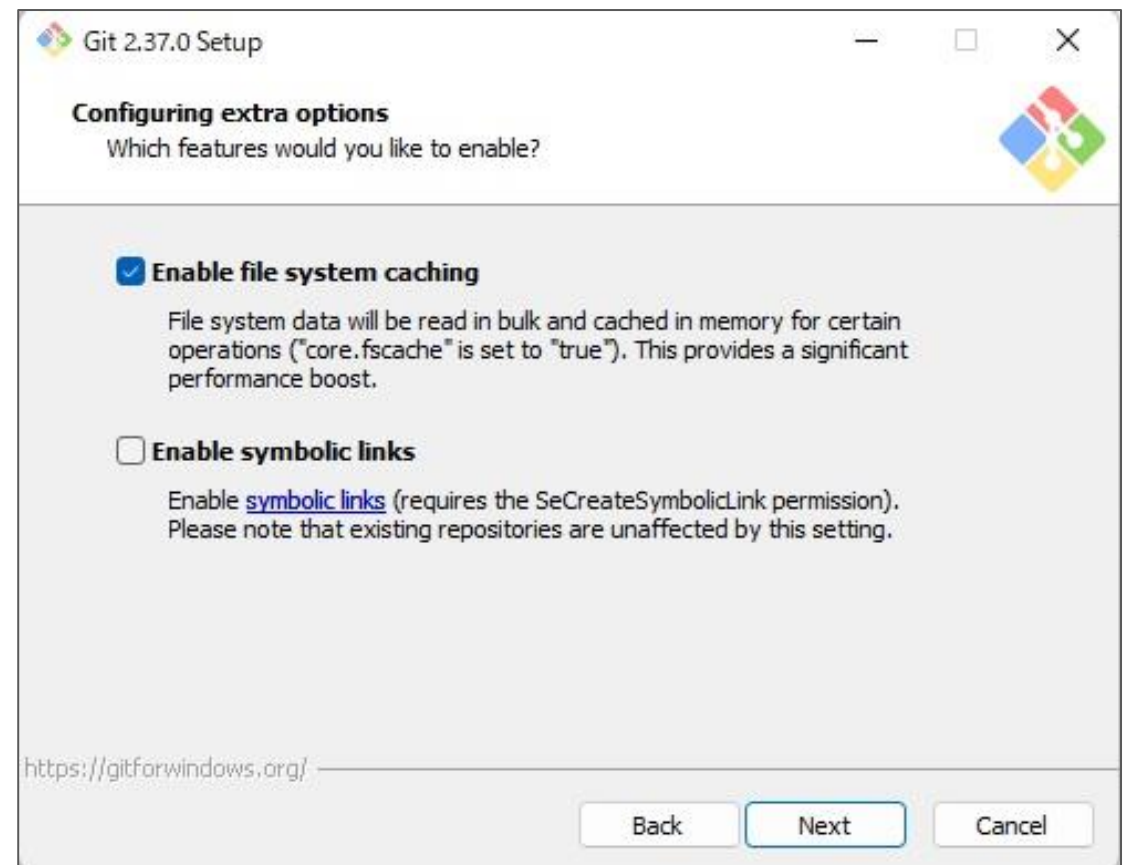
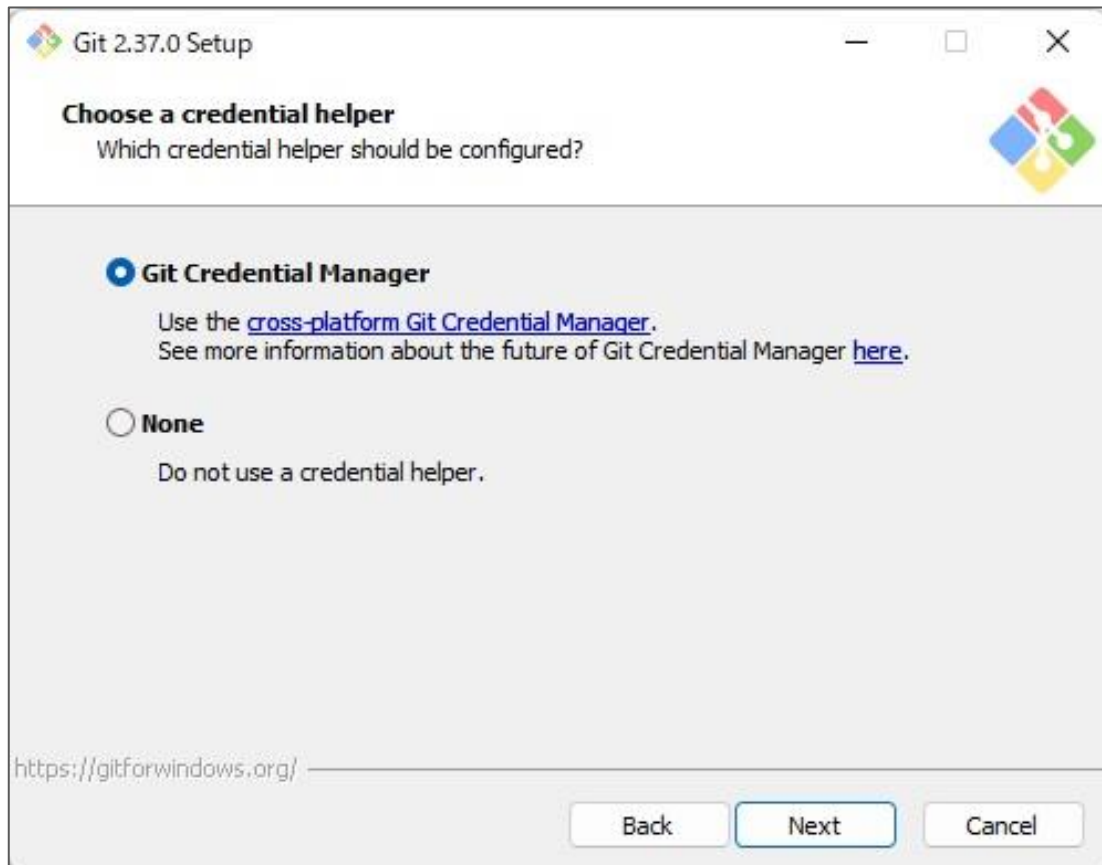
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



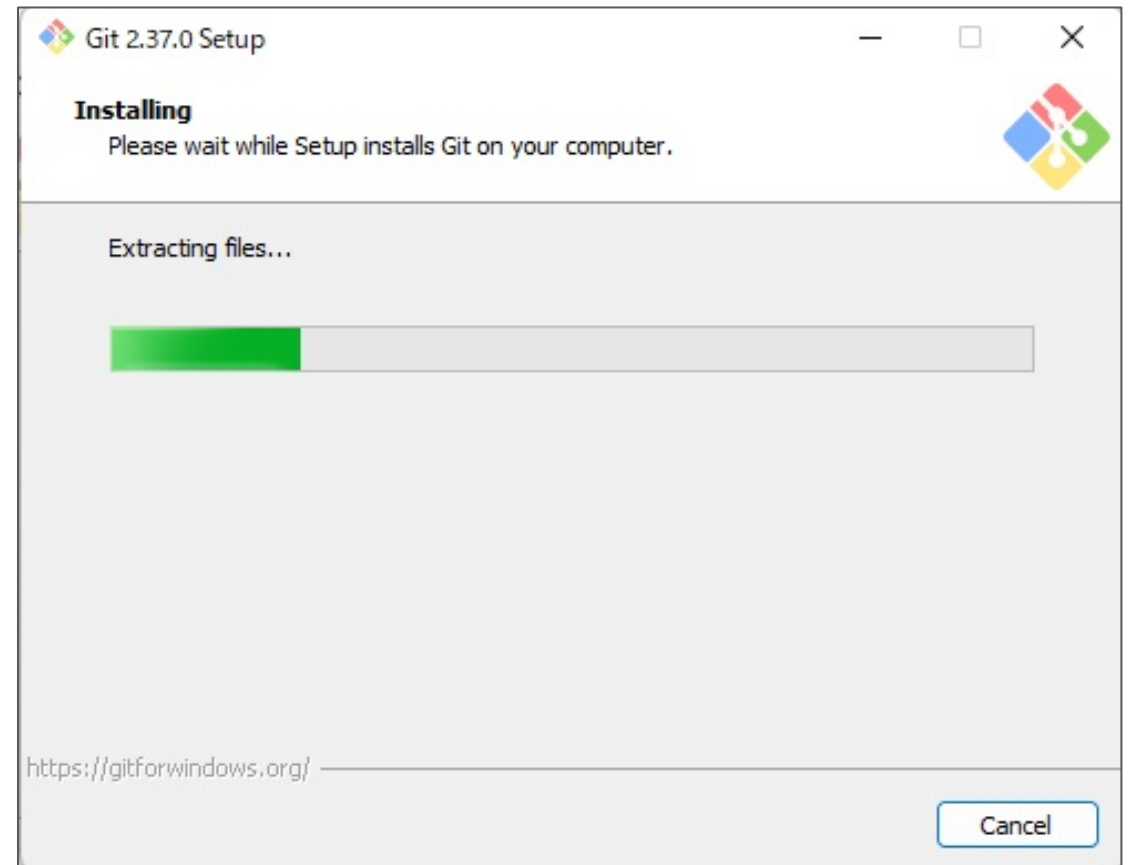
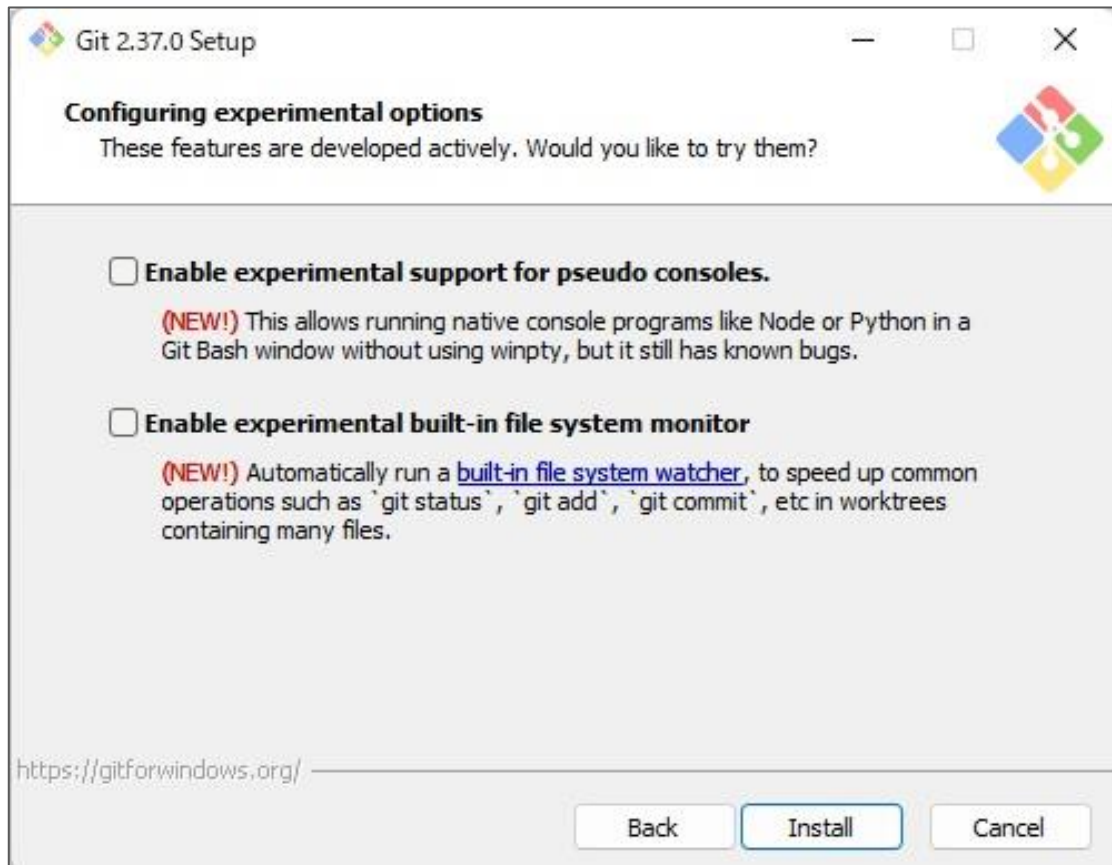
Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



Gitツールのインストール (Windows)

- デフォルト変更せずに“Next”をクリック



Gitツールのインストール (Windows)

- “Finish”が出ればインストール終了



```
PS C:\Users\Yosono> git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
          [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
          [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
          [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
          [--super-prefix=<path>] [--config-env=<name>=<envvar>]
          <command> [<args>]
```

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)

clone	Clone a repository into a new directory
init	Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)

add	Add file contents to the index
mv	Move or rename a file, a directory, or a symlink
restore	Restore working tree files
rm	Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)

bisect	Use binary search to find the commit that introduced a bug
diff	Show changes between commits, commit and working tree, etc
grep	Print lines matching a pattern
log	Show commit logs
show	Show various types of objects
status	Show the working tree status

Gitツールの動作確認（Windows）

- PowerShellで以下のコマンドを実行

```
> git
```

実行例

```
PS C:\Users\Yosono> git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
    [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
    [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
    [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
    [--super-prefix=<path>] [--config-env=<name>=<envvar>]
    <command> [<args>]
```

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)

clone	Clone a repository into a new directory
init	Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)

add	Add file contents to the index
mv	Move or rename a file, a directory, or a symlink
restore	Restore working tree files
rm	Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)

bisect	Use binary search to find the commit that introduced a bug
diff	Show changes between commits, commit and working tree, etc
grep	Print lines matching a pattern
log	Show commit logs
show	Show various types of objects
status	Show the working tree status

Gitツールのインストール (Mac)

- Homebrewを使ってインストール

```
% brew install git
```

- 動作確認

```
% git --version
```

出力例

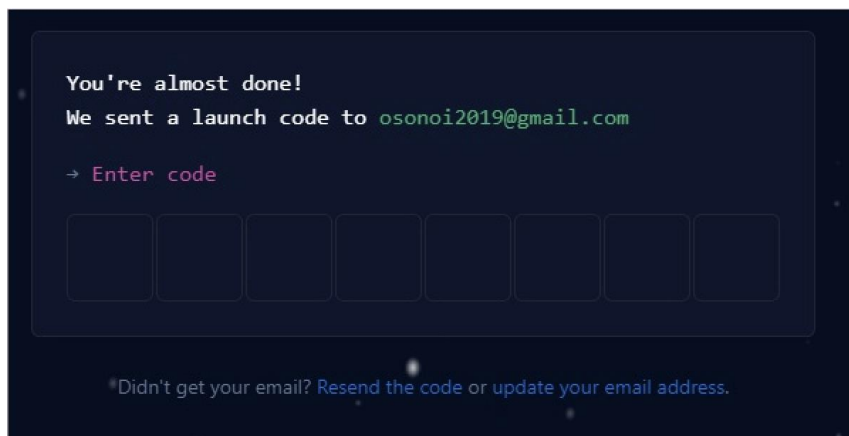
```
[~ % git --version  
git version 2.37.0
```

GitHubアカウントの作成

- <https://github.co.jp/>の右上で“Sign up”を選択

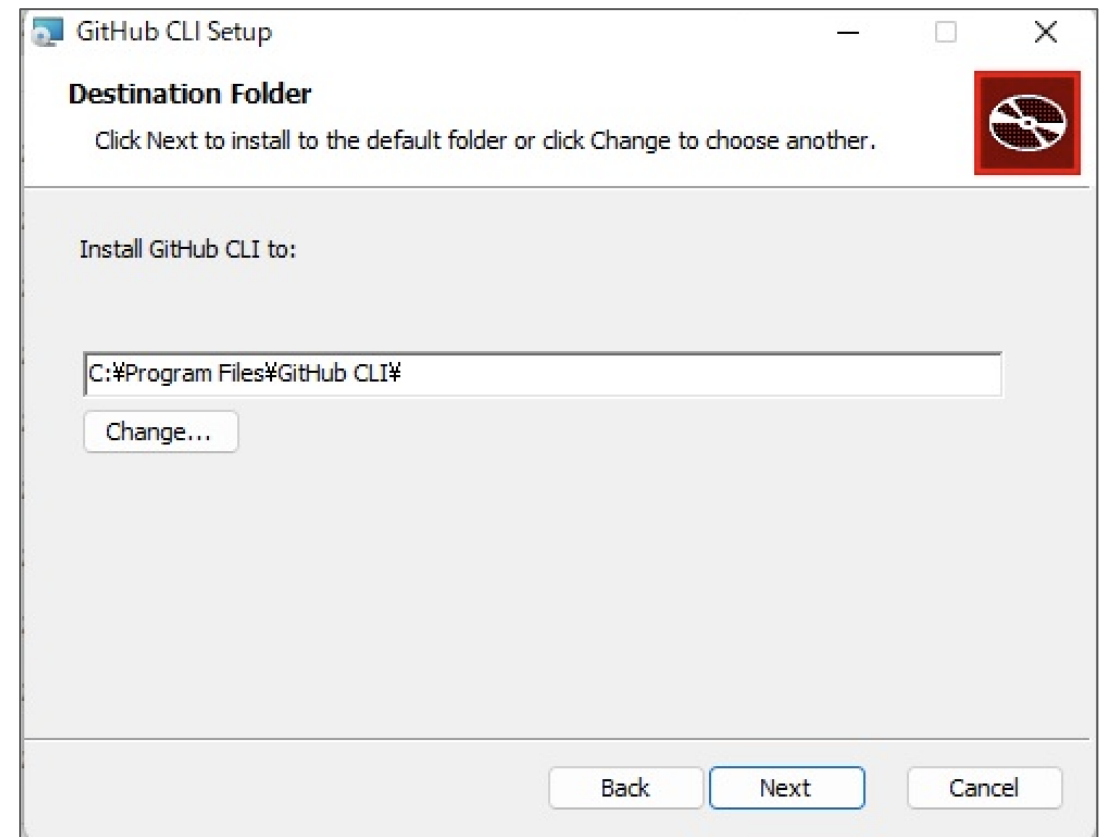
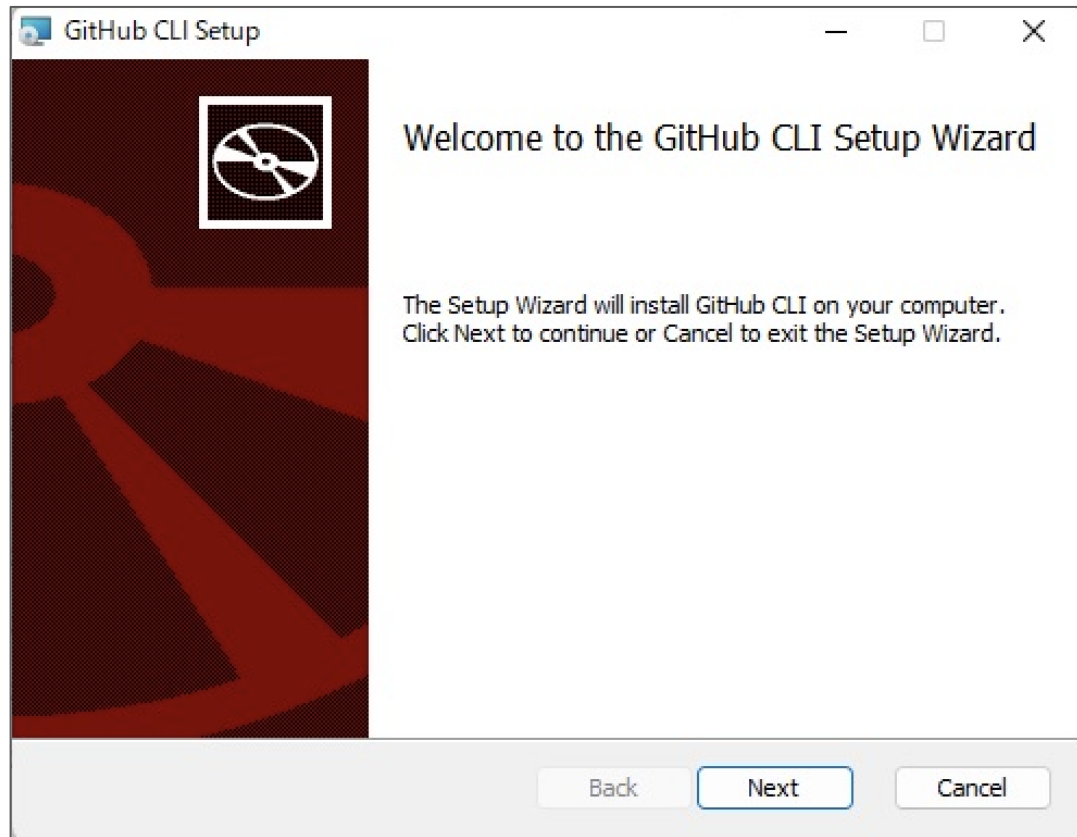


- メールアドレス、パスワード、usernameを入力
- 8桁のコードが上記メールアドレスに送付されるのでそれを入力して登録を進める



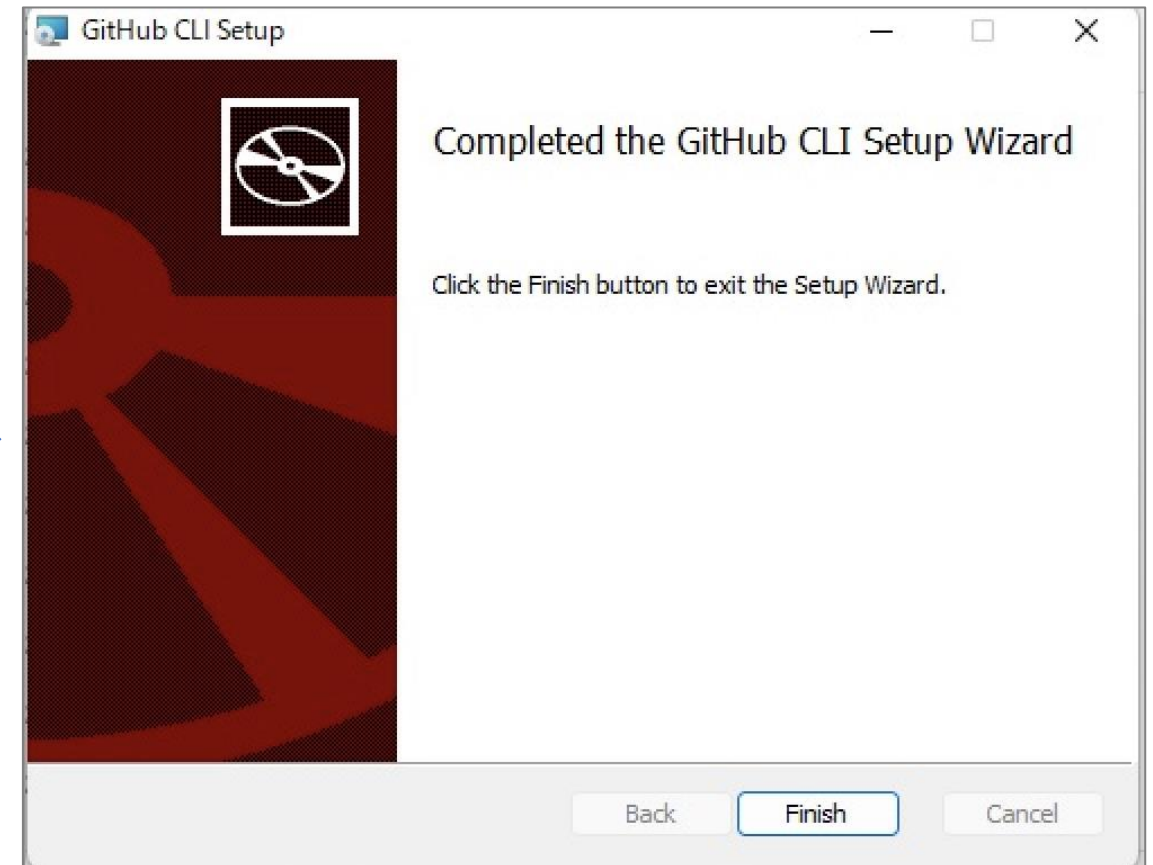
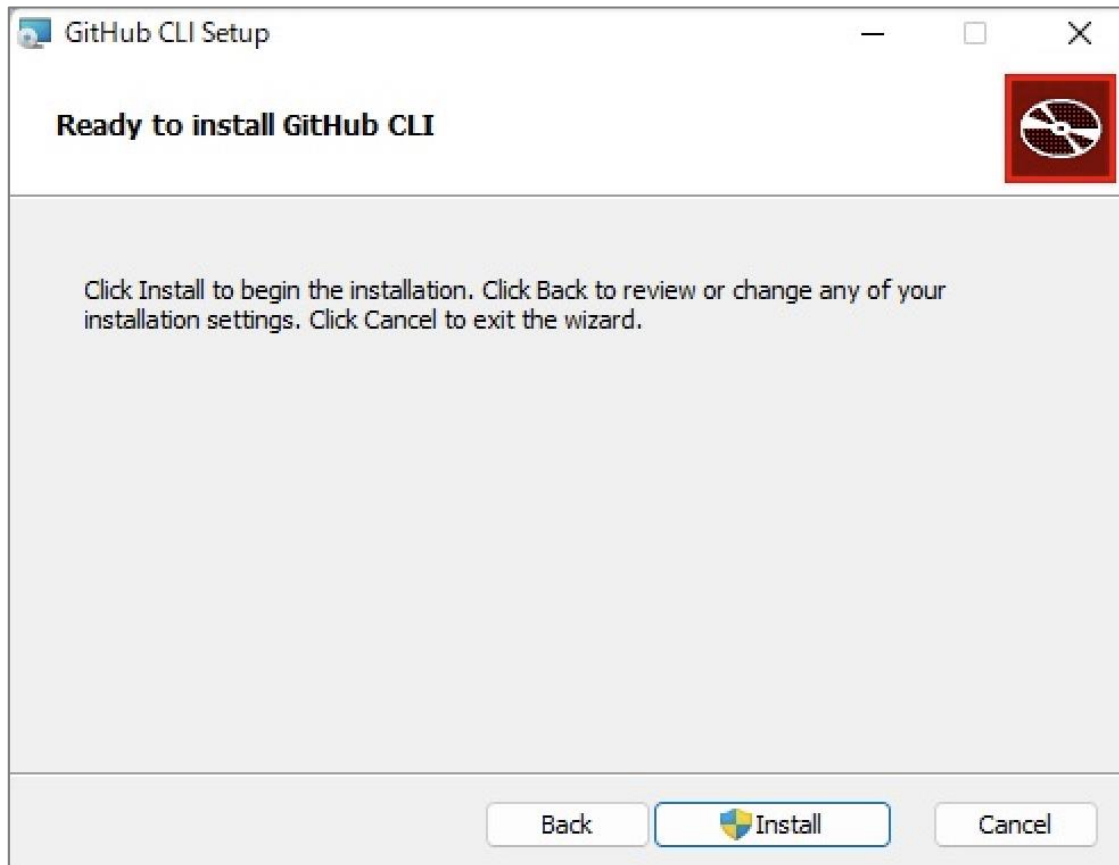
Github CLIのインストール (Windows)

- <https://github.com/cli/cli/releases/tag/v2.14.2>
から [gh_2.14.2_windows_amd64.msi](#)
をダウンロードして実行



Github CLIのインストール (Windows)

- 基本的にはデフォルトで進める



Github CLIのインストール (Windows)

- インストールの確認

```
> gh --version
```

実行例

```
PS C:\Users\osono> gh --version  
gh version 2.14.2 (2022-07-14)  
https://github.com/cli/cli/releases/tag/v2.14.2  
PS C:\Users\osono> _
```


Github CLIのインストール (Mac)

- Homebrewでインストール

```
% brew install gh
```

- インストール確認

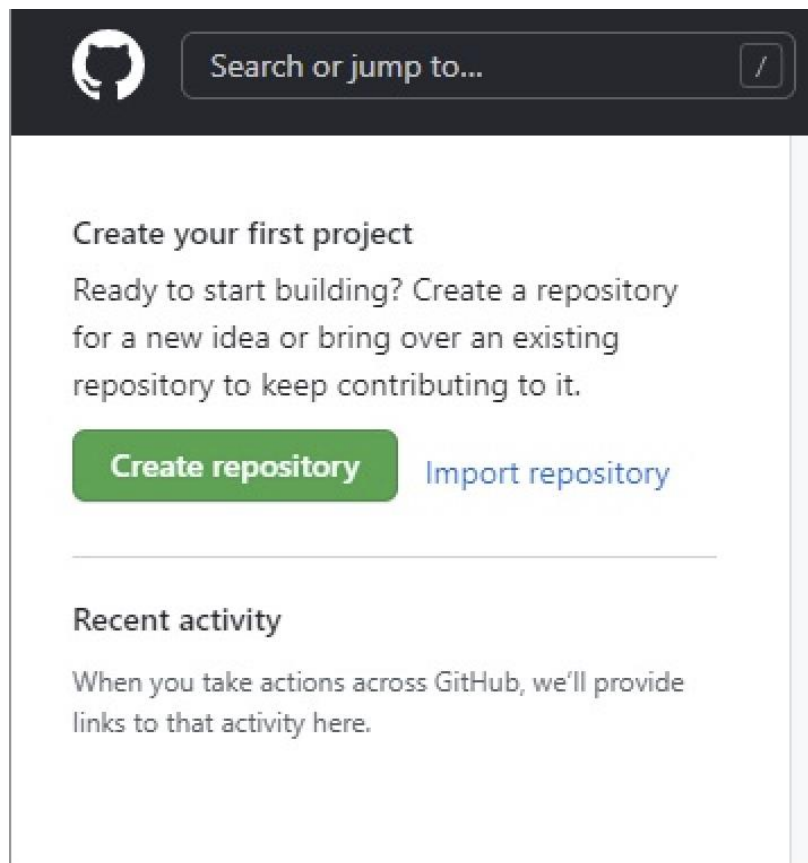
```
% gh --version
```

実行例

```
[~ % gh --version  
gh version 2.14.2 (2022-07-14)  
https://github.com/cli/cli/releases/tag/v2.14.2  
~ % █
```

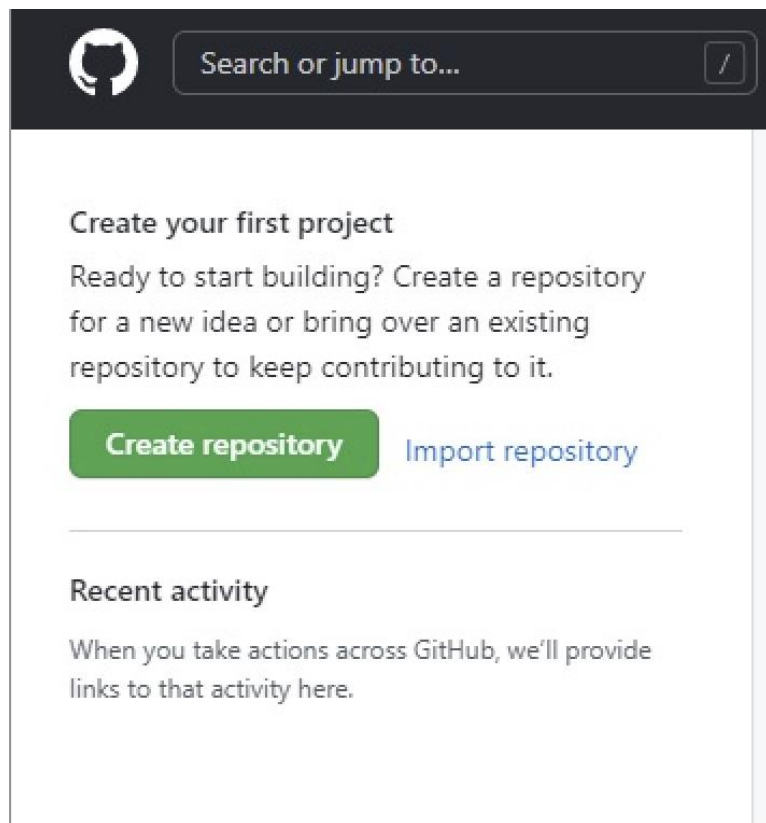
GitHub演習

- レポジトリの作成
- ブラウザでGitHubにログイン、“Create repository”をクリック



GitHub演習

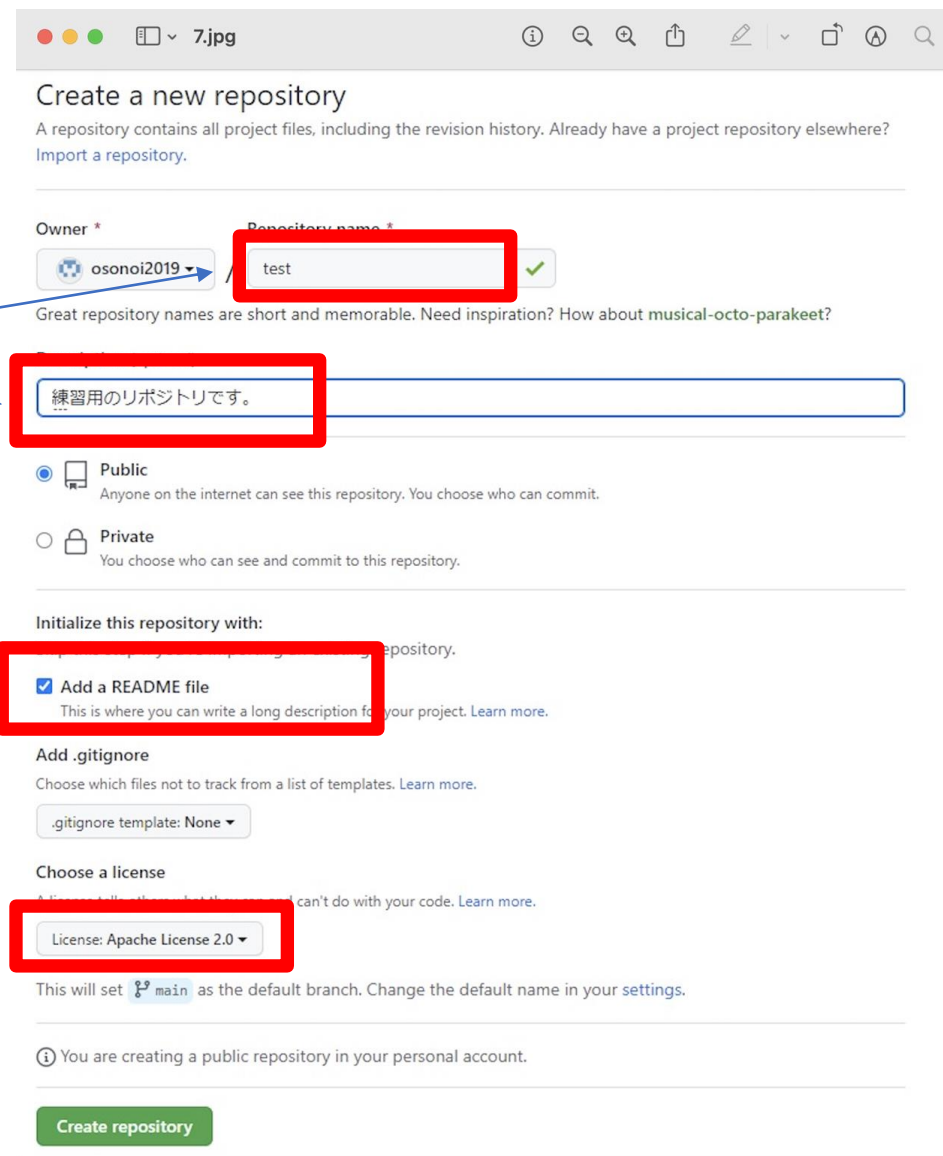
- レポジトリの作成
- ブラウザでGitHubにログイン、
- "Create repository"をクリック



任意の名前、
コメントを入
れる

チェックを入
れる

ライセンスも入
れて
おくことを推奨



GitHub演習

- 以下のようにレポジトリが作成されるのを確認

The screenshot shows the GitHub interface for a repository named 'osonoi2019/test'. The repository is public and has a README file. The README content is as follows:

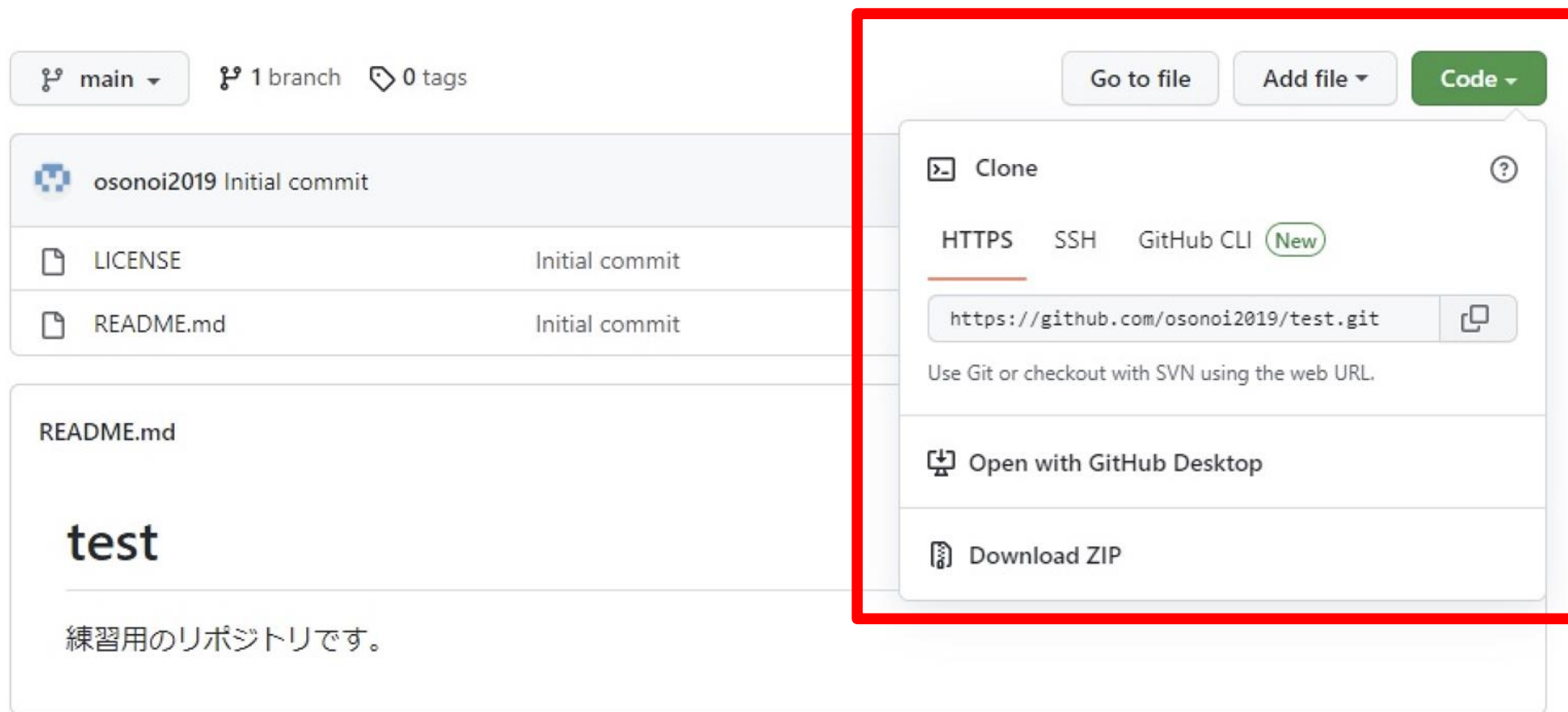
```
test

練習用のリポジトリです。
```

The repository page includes a navigation bar with links to Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The repository details section shows the initial commit (52133ba) and the README file. The right sidebar contains information about the repository, including the number of stars (0), watchers (1), and forks (0). The footer of the page displays the GitHub logo and copyright information for 2022.

GitHub演習

- レポジトリのアドレスを取得しておく



The screenshot shows a GitHub repository page for 'osonoi2019'. The repository has 1 branch and 0 tags. The 'main' branch is selected. The repository contains two files: 'LICENSE' and 'README.md', both from the 'Initial commit'. The 'README.md' file is expanded, showing the text 'test' and '練習用のリポジトリです。'. A red box highlights the 'Code' dropdown menu, which is open and shows the following options:

- Clone (with a question mark icon)
- HTTPS (selected, with a 'New' label)
- SSH
- GitHub CLI
- The URL `https://github.com/osonoi2019/test.git` is displayed with a copy icon.
- Use Git or checkout with SVN using the web URL.
- Open with GitHub Desktop
- Download ZIP

GitHub演習

- 事前にブラウザでGitHubにログインしてください
- コマンドラインからGitHubにログイン
- `% gh auth login`
- 上記コマンド入力後右の水色のように入力して

```
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? HTTPS
[? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 717A-99A1
Press Enter to open github.com in your browser...
```

ブラウザにこの文字
を
入力します。

- 最後にEnterを押すとブラウザが立ち上がります。
- ブラウザが立ち上がったら上記文字を入力します。
- コマンドラインにlogged in ...が出力され入力可能になる

GitHub演習

- ソースコードのclone (GitHubからPCにコピー)

```
% git clone https://github.com/(ご自分のアカウント)/test.git  
% cd test
```

- ソースコードがクローン (コピー) され自分のPCに

出力例

```
[~ % git clone https://github.com/osonoi2019/test.git  
Cloning into 'test'...  
remote: Enumerating objects: 8, done.  
remote: Counting objects: 100% (8/8), done.  
remote: Compressing objects: 100% (7/7), done.  
remote: Total 8 (delta 0), reused 3 (delta 0), pack-reused 0  
Receiving objects: 100% (8/8), 5.46 KiB | 5.46 MiB/s, done.  
[~ % cd test  
test % █
```

GitHub演習

- ソースコードの編集
- PC上のエディターを使ってREADME.mdを編集
- Macはテキストエディット、Windowsはメモ帳など
- VS codeを推奨
- gitコマンドで編集を確認

```
% git status
```

出力例

```
[test % git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

README.mdが編集
されたことを表示

GitHub演習

- ソースコードの編集
- 編集したファイルをコミット対象にする

```
% git add README.md
```

- 次にコメントをつけてコミットする

```
% git commit -m "README.mdを編集"
```

- コミット
 - 追加・変更したファイルをGitに登録する
 - ソフトのテストなどが終わって確認できたら行う

GitHub演習

- 前項目の操作でPC上のソースコードが変更されたがGitHub上のコードは修正されていない
- 以下のコマンドでGitHub上のソースコードを修正

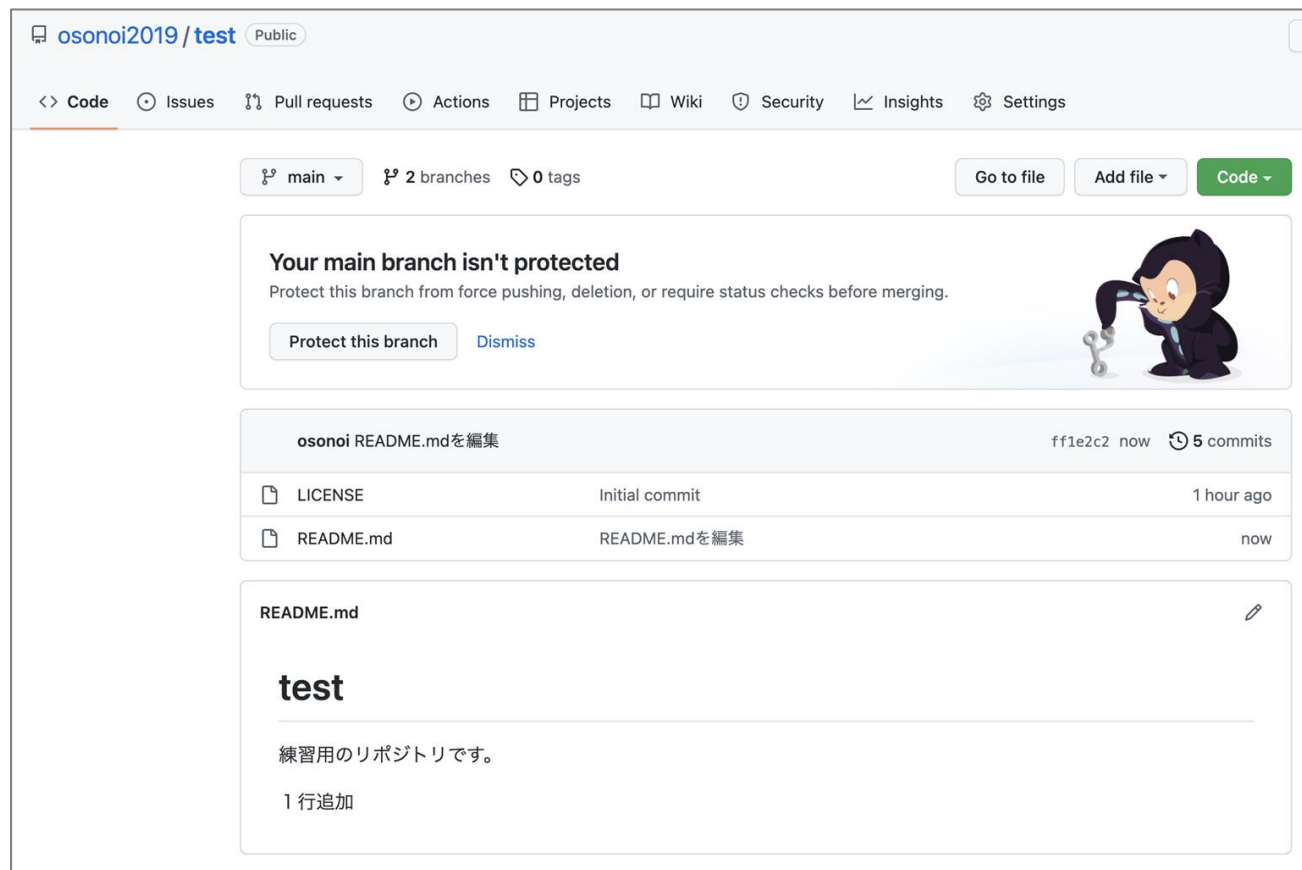
```
% git push
```

実行例

```
[test % git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 350 bytes | 350.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/osonoi2019/test.git
  9f03733..3e472a7  main -> main
```

GitHub演習

- GitHubのページで編集箇所を確認



GitHubについて補足

- 今回の演習ではGitHubのmainブランチに直接変更を加えたが実際の案件では別ブランチを作成してそこを変更、全体管理している人にプルリクエストを送ってMainブランチを変更したりなどチームで管理する方法が色々あるので実際の案件に入ったらそのやり方に従う必要がある

第1章 まとめ

GitおよびGitHubを学習し

- GitHubの概要として以下を学習した
 - Gitとはソフトウェアの変更履歴管理・バージョンを管理するツールである
 - GitHubとはGitを利用した開発者を支援するWebサービスで、バージョン管理コラボレーションのためのコードホスティングプラットフォームである
- GitHubの基本的な使い方として、以下を学習した
 - GitHub上でのプログラムの管理、共有
 - ローカル（PC上の）環境との同期
 - GitHub上にリポジトリを作成し、ローカルにクローン、編集後GitHubにプッシュ（アップロード）

JavaScriptフレームワークによるWebプログラミング
第3回Node.js実習1

第1章 GitHub概要および実習

終わり

JavaScriptフレームワークによるWebプログラミング

第3回 Nodejs実習1

第2章 nodejsでサイトを作成

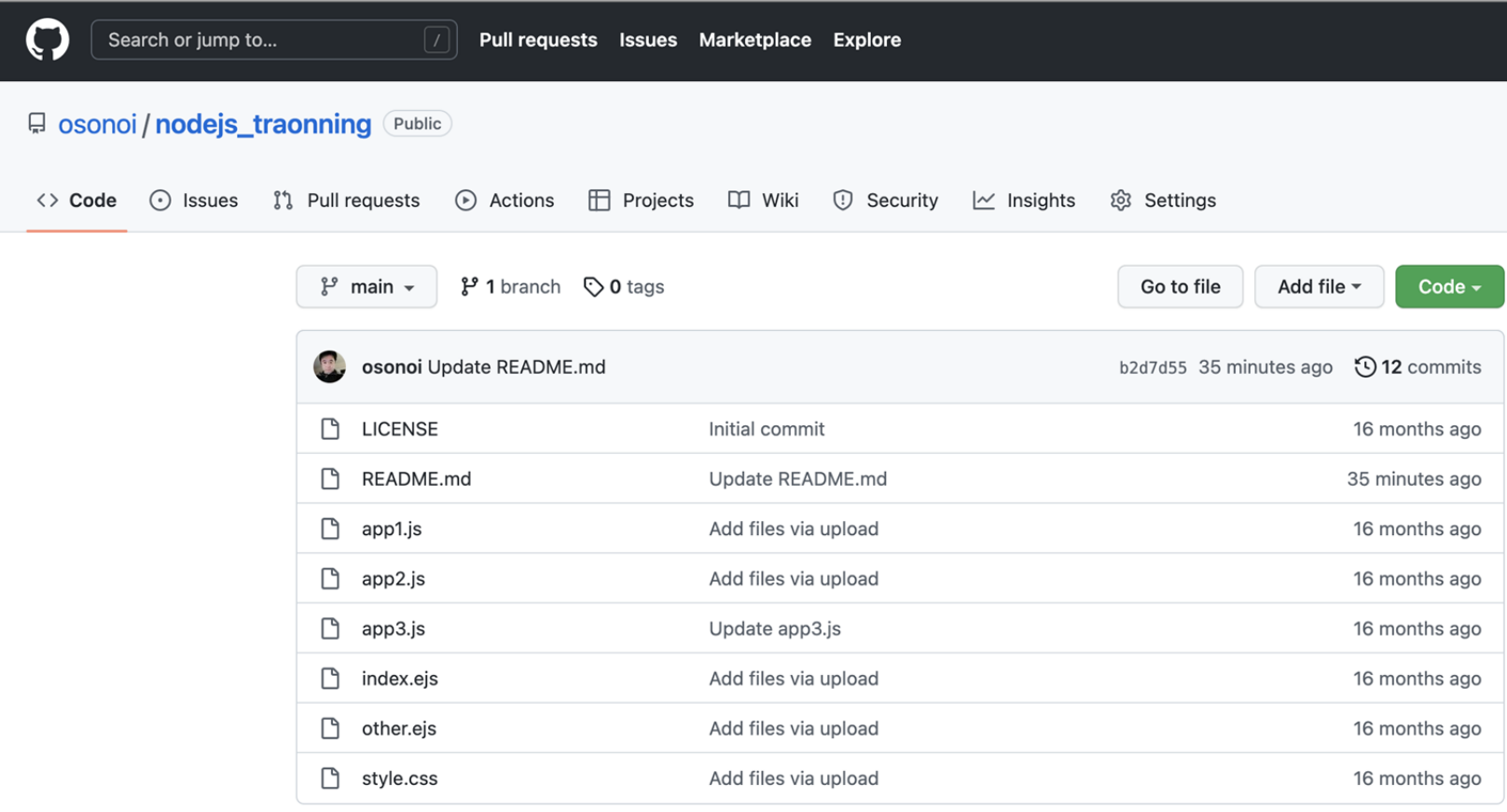
第2章 学習目標

- GitHubからソースコードをダウンロードしてきてNode.jsのサイトをローカルで立ち上げることができる

Node.jsでサイトを立ち上げる

- ブラウザーで以下のサイトを表示確認

https://github.com/osonoi/nodejs_training



Search or jump to... Pull requests Issues Marketplace Explore

osonoi / nodejs_training Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

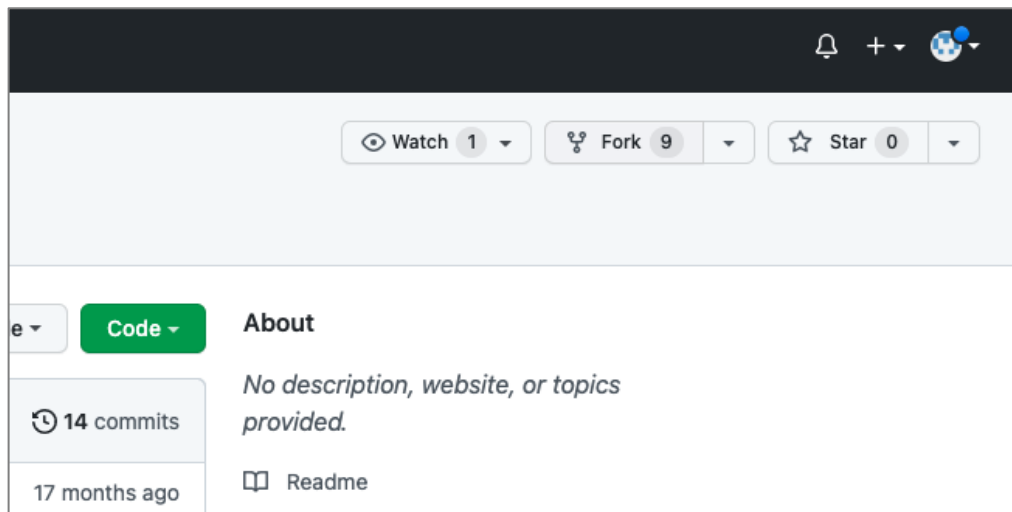
main 1 branch 0 tags Go to file Add file Code

osonoi Update README.md b2d7d55 35 minutes ago 12 commits

LICENSE	Initial commit	16 months ago
README.md	Update README.md	35 minutes ago
app1.js	Add files via upload	16 months ago
app2.js	Add files via upload	16 months ago
app3.js	Update app3.js	16 months ago
index.ejs	Add files via upload	16 months ago
other.ejs	Add files via upload	16 months ago
style.css	Add files via upload	16 months ago

Node.jsでサイトを立ち上げる


- リポジトリを自分のアカウントにフォーク
GitHubにログイン画面右上の"Fork"をクリック



Create a new fork

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. [View existing forks.](#)

Owner * Repository name *

 osonoi2019

By default, forks are named the same as their parent repository. You can customize the name to distinguish it further.

Description (optional)

📘 You are creating a fork in your personal account.

Create fork

自分のアカウント名

Node.jsでサイトを立ち上げる

- 以下のコマンドでGitHubのソースコードをローカルにコピー (Clone)

```
% git clone https://github.com/(アカウント名)/nodejs_training.git
```

(Windows環境でも同じコマンド)

- CDでソースコードの中に入りプログラムを実行

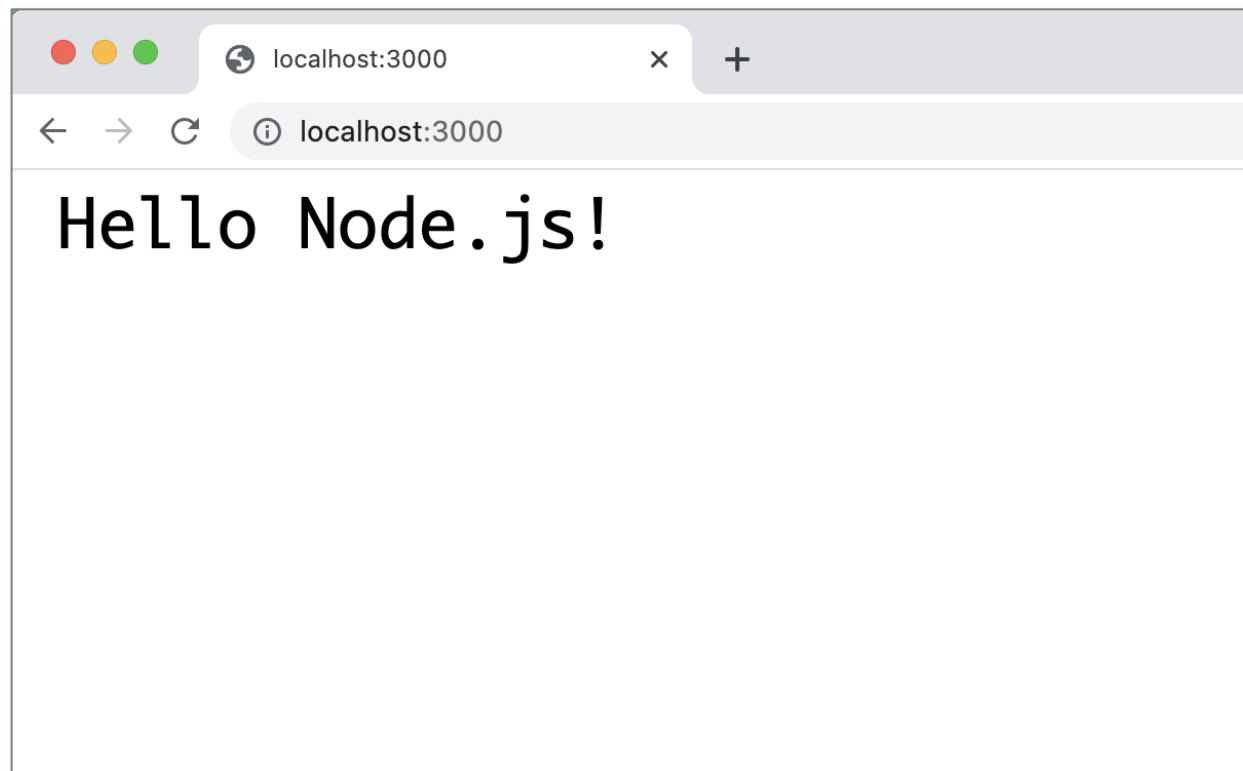
```
% cd nodejs_training
```

```
% node app1.js
```

- ブラウザーを開いて <http://localhost:3000/> を確認

Node.jsでサイトを立ち上げる

- ブラウザで動作を確認



* ポイント : Nodejsだけで 1 Webサーバが実現できる

Node.jsでサイトを立ち上げる

- プログラムを停める場合はコマンド画面で “Control + C” を押す

```
[nodejs_training % node app1.js  
^C  
nodejs_training % █
```

Node.jsでサイトを立ち上げる

- ソースコードを確認

```
1  const http = require('http');  
2  
3  var server = http.createServer(  
4    (request, response) => {  
5      response.end('Hello Node.js!');  
6    }  
7  );  
8  server.listen(3000);  
9
```

サーバオブジェクトを作成

サーバにアクセスがあったときの動作
を記述

ポート3000で待ち受け状態にする

次のソースコードをローカルで実行

```
% node app2.js
```

ブラウザを開いて <http://localhost:3000/> を確認



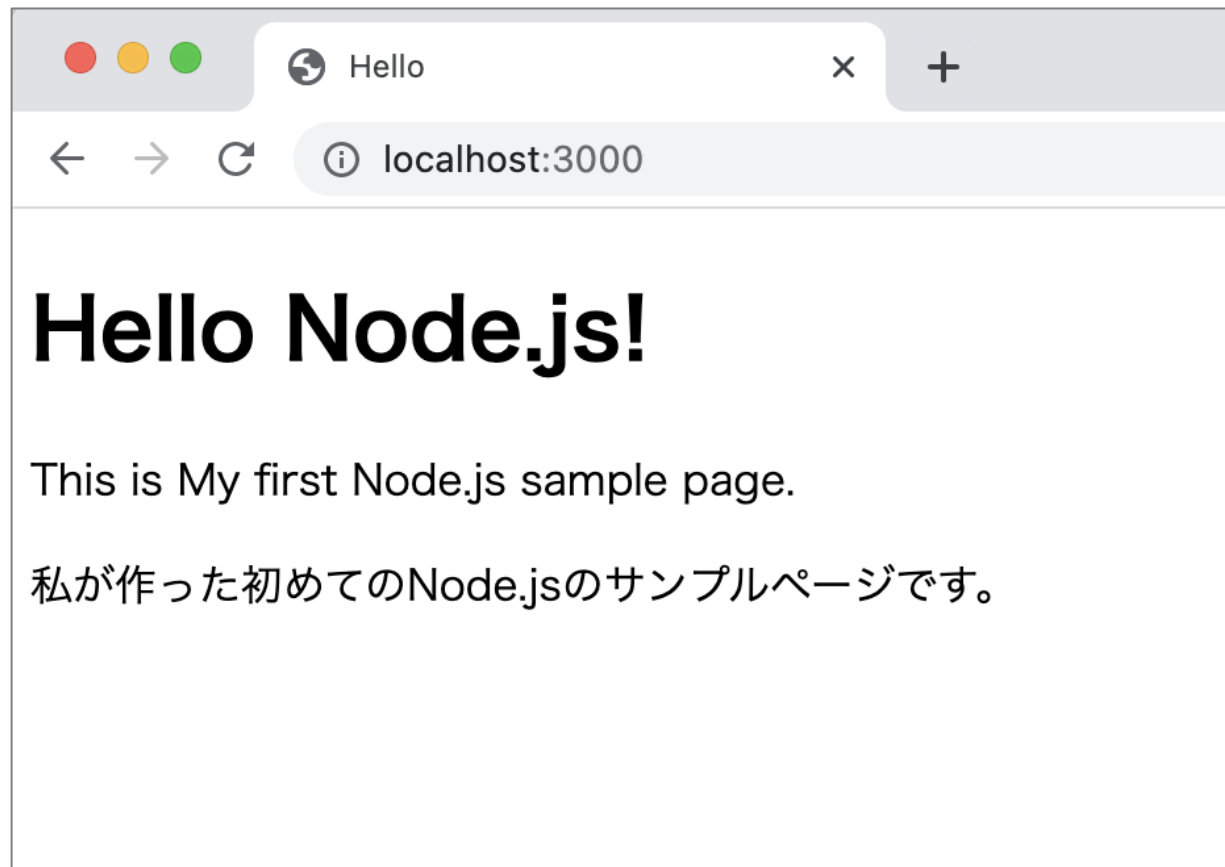
Node.jsでサイトを立ち上げる

- App2.jsのソースコード

```
1  const http = require('http');
2
3  var server = http.createServer(
4    (request, response) => {
5      response.setHeader('Content-Type', 'text/html');
6      response.write('<!DOCTYPE html><html lang="ja">');
7      response.write('<head><meta charset="utf-8">');
8      response.write('<title>Hello</title></head>');
9      response.write('<body><h1>Hello Node.js!</h1>');
10     response.write('<p>This is Node.js sample page.</p>');
11     response.write('<p>これは、Node.jsのサンプルページです。</p>', 'utf8');
12     response.write('</body></html>');
13     response.end();
14   }
15 );
16
17 server.listen(3000);
18 console.log('Server start!');
19
```


Node.jsでサイトを立ち上げる実習

- ページ内のメッセージを修正してみる



第2章 まとめ

- 以下の内容を学習した。
 - GitHubからソースコードをダウンロード
予め作成されていたGitHub上のnodejsプログラムをClone
 - Node.jsのサイトをローカルで立ち上げる
ローカル環境でNode.jsプログラムを立ち上げる
ブラウザーで動作を確認

第3回 まとめ

- ・ GitHubの概要および基本的な使い方として、以下を学習した
 - GitHub上でのプログラムの管理（レポジトリの作成、ブランチの使用方法）
 - GitHub上にあるソースコードのコピー、編集
- ・ GitHubからソースコードをダウンロードし、Node.jsのサイトを立ち上げた。

JavaScriptフレームワークによるWebプログラミング
第3回 Nodejs実習1

第2章

Node.js実習1

終わり