VSCodeからGitHubへの接続・プッシュ完全ガイド

目次

1. 事前準備
2. Gitの初期設定
3. ローカルリポジトリの作成
4. GitHubリモートリポジトリの作成
5. VSCodeでの認証設定
6. ファイルの追加とコミット
7. リモートリポジトリへのプッシュ
8. トラブルシューティング

事前準備

必要なソフトウェアのインストール

Visual Studio Code

- 公式サイト（https://code.visualstudio.com/）からダウンロードしてインストールしてください
- Windows11・macOS共通の手順です

Git Gitとは、ファイルのバージョン管理を行うためのシステムです。コードの変更履歴を追跡し、複数の開発者が協力して作業できるようにします。

Windows11の場合：

1. Git公式サイト（https://git-scm.com/）にアクセス
2. 「Download for Windows」をクリック
3. ダウンロードされたインストーラーを実行
4. インストール時は基本的にデフォルト設定で問題ありません

macOSの場合： macOSには通常Gitがプリインストールされていますが、最新版を使用したい場合：

方法1（Homebrewを使用）：

```bash
brew install git
```

方法2（公式インストーラー）：

1. Git公式サイト（https://git-scm.com/）にアクセス
2. 「Download for macOS」をクリック
3. インストーラーをダウンロードして実行

GitHubアカウント

- GitHub（https://github.com/）でアカウントを作成してください
- 無料アカウントで十分です

Gitの初期設定

Gitを使用する前に、あなたの身元情報を設定する必要があります。これにより、コミット（変更の記録）に誰が変更を行ったかが記録されます。

ターミナル/コマンドプロンプトでの設定

Windows11の場合：

1. スタートメニューを右クリック
2. 「Windows PowerShell」または「コマンドプロンプト」を選択
3. 以下のコマンドを実行（あなたの情報に置き換えてください）：

```bash
git config --global user.name "あなたの名前"
git config --global user.email "あなたのメールアドレス"
```

macOSの場合：

1. 「アプリケーション」→「ユーティリティ」→「ターミナル」を開く
2. 以下のコマンドを実行（あなたの情報に置き換えてください）：

```bash
git config --global user.name "あなたの名前"
git config --global user.email "あなたのメールアドレス"
```

注意事項：

- メールアドレスはGitHubに登録したものと同じものを使用してください
- 名前は本名でなくても構いませんが、一貫性を保つことが重要です

設定の確認

以下のコマンドで設定が正しく行われたかを確認できます：

```bash
git config --global --list
```

ローカルリポジトリの作成

リポジトリとは、プロジェクトのファイルと変更履歴を保存する場所です。ローカルリポジトリは、あなたのコンピューター上にあるリポジトリのことです。

VSCodeでプロジェクトフォルダを開く

1. VSCodeを起動
2. 「ファイル」メニュー → 「フォルダーを開く」を選択
3. プロジェクトを作成したいフォルダを選択または新規作成

Gitリポジトリの初期化

方法1：VSCodeの統合ターミナルを使用

1. VSCode内で「Ctrl + 」（Windows）または「Cmd + 」（Mac）を押してターミナルを開く
2. 以下のコマンドを実行：

```bash
git init
```

このコマンドにより、現在のフォルダにGitリポジトリが作成されます。

方法2：VSCodeのGUIを使用

1. VSCodeのサイドバーで「ソース管理」アイコン（分岐のようなアイコン）をクリック
2. 「リポジトリを初期化」ボタンをクリック

.gitignoreファイルの作成

.gitignoreファイルは、Gitに追跡させたくないファイルやフォルダを指定するファイルです。

1. VSCodeでプロジェクトフォルダに `.gitignore` ファイルを作成
2. 以下の内容を記載（プロジェクトの種類に応じて調整してください）：

```
# 依存関係
node_modules/
.npm

# ログファイル
*.log

# 環境変数ファイル
.env
.env.local

# OS固有ファイル
.DS_Store
Thumbs.db

# エディタ固有ファイル
.vscode/
.idea/

# ビルド出力
dist/
build/
```

GitHubリモートリポジトリの作成

リモートリポジトリとは、インターネット上（この場合GitHub）に保存されるリポジトリです。ここにコードをアップロード（プッシュ）することで、バックアップや他の人との共有が可能になります。

GitHubでの新規リポジトリ作成

1. GitHubにログイン
2. 右上の「+」アイコンをクリック → 「New repository」を選択
3. 以下の情報を入力：
   - **Repository name**: プロジェクト名を入力
   - **Description**: プロジェクトの説明（任意）
   - **Public/Private**: 公開するか非公開にするかを選択
   - **Initialize this repository with a README**: チェックを外す（ローカルで既にファイルがある場合）
4. 「Create repository」をクリック

リモートリポジトリのURLをコピー

リポジトリ作成後、画面に表示されるHTTPS URLをコピーしてください。 例：`https://github.com/ユーザー名/リポジトリ名.git`

VSCodeでの認証設定

GitHubとの通信には認証が必要です。VSCodeでは複数の認証方法をサポートしています。

GitHub拡張機能のインストール

1. VSCodeの拡張機能タブ（四角いアイコン）をクリック
2. 「GitHub」で検索
3. 「GitHub Pull Requests and Issues」をインストール

GitHubアカウントとの連携

1. VSCode下部のステータスバーでアカウントアイコンをクリック
2. 「Turn on Settings Sync...」を選択
3. 「Sign in with GitHub」を選択
4. ブラウザが開くので、GitHubアカウントでログイン
5. VSCodeへのアクセスを許可

Personal Access Token（代替方法）

GitHub拡張機能での認証がうまくいかない場合は、Personal Access Tokenを使用できます。

1. GitHubの「Settings」→「Developer settings」→「Personal access tokens」→「Tokens (classic)」
2. 「Generate new token (classic)」をクリック
3. 適切な権限（repo、workflow等）を選択
4. トークンを生成してコピー（画面を閉じる前に必ずコピーしてください）

ファイルの追加とコミット

コミットとは、ファイルの変更を記録として保存することです。スナップショットのようなもので、後で特定の時点の状態に戻ることができます。

ファイルの作成と編集

1. VSCodeでプロジェクトファイルを作成・編集
2. 例として、`README.md` ファイルを作成：

```markdown
# プロジェクト名

このプロジェクトについての説明をここに書きます。

## 使用方法

使用方法について説明します。
```

変更の確認

1. VSCodeのソース管理タブ（分岐アイコン）を開く
2. 変更されたファイルが「Changes」セクションに表示されます
3. ファイル名をクリックすると、変更内容の差分が表示されます

ステージングエリアに追加

ステージングエリアとは、次のコミットに含める変更を準備する場所です。

方法1：VSCodeのGUIを使用

1. 「Changes」セクションで、追加したいファイルの「+」アイコンをクリック
2. ファイルが「Staged Changes」セクションに移動します

方法2：ターミナルを使用

```bash
git add ファイル名        # 特定のファイルを追加
git add .                 # すべての変更を追加
```

コミットの実行

1. ステージングエリアにファイルを追加後、コミットメッセージを入力
2. メッセージ入力欄に意味のあるメッセージを記入（例：「初期プロジェクトファイルを追加」）
3. 「Ctrl + Enter」（Windows）または「Cmd + Enter」（Mac）でコミット実行

良いコミットメッセージの例：

- 「ユーザー認証機能を追加」
- 「バグ修正：ログイン時のエラーを解決」
- 「README.mdファイルを更新」

リモートリポジトリへのプッシュ

プッシュとは、ローカルリポジトリの変更をリモートリポジトリ（GitHub）に送信することです。

リモートリポジトリの追加

初回のみ、リモートリポジトリの場所をローカルリポジトリに教える必要があります。

VSCodeターミナルで実行：

```bash
git remote add origin https://github.com/ユーザー名/リポジトリ名.git
```

`origin`とは、リモートリポジトリの標準的な名前です。複数のリモートリポジトリがある場合に区別するために使用します。

ブランチ名の確認と設定

現代のGitでは、デフォルトブランチ名として`main`が推奨されています。

現在のブランチ名を確認：

```bash
git branch
```

ブランチ名を`main`に変更（必要に応じて）：

```bash
git branch -M main
```

初回プッシュの実行

方法1：VSCodeのGUIを使用

1. ソース管理タブで「...」メニューをクリック
2. 「プッシュ」を選択
3. 初回の場合、リモートブランチとの関連付けを確認するダイアログが表示されるので「OK」をクリック

方法2：ターミナルを使用

```bash
git push -u origin main
```

`-u`オプションは、ローカルの`main`ブランチとリモートの`main`ブランチを関連付けるためのものです。初回のみ必要で、以降は`git push`だけでプッシュできます。

認証の実行

プッシュ時に認証が求められます：

Personal Access Tokenを使用する場合：

- ユーザー名：GitHubのユーザー名
- パスワード：生成したPersonal Access Token

GitHub拡張機能で認証済みの場合：

- 自動的に認証が処理されます

プッシュの成功確認

1. GitHubのリポジトリページをブラウザで確認
2. ファイルが正しくアップロードされているかを確認
3. コミットメッセージが表示されているかを確認

継続的な開発フロー

初回設定が完了した後は、以下のフローで開発を進めます：

日常的な作業フロー

1. **ファイルの編集**：VSCodeでコードを編集
2. **変更の確認**：ソース管理タブで変更内容を確認
3. **ステージング**：変更をステージングエリアに追加
4. **コミット**：意味のあるメッセージでコミット
5. **プッシュ**：リモートリポジトリに変更を送信

プッシュの実行（2回目以降）

初回設定後は、以下のコマンドで簡単にプッシュできます：

VSCodeのGUIを使用：

- ソース管理タブの「↑」アイコンをクリック

ターミナルを使用：

```
bash
```

```
git push
```

トラブルシューティング

よくある問題と解決方法

問題1：「fatal: not a git repository」エラー

原因：Gitリポジトリが初期化されていない **解決方法**：プロジェクトフォルダで`git init`を実行

問題2：認証エラー

原因：GitHubの認証情報が正しくない **解決方法**：

- Personal Access Tokenを再生成
- VSCodeのGitHub拡張機能で再認証
- 認証情報をクリア後、再入力

問題3：「Updates were rejected」エラー

原因：リモートリポジトリに新しい変更があり、ローカルが古い状態 **解決方法**：

```bash
git pull origin main  # リモートの変更を取得
git push              # 再度プッシュ
```

問題4：ファイルが追跡されない

原因：.gitignoreファイルで除外されている **解決方法**：.gitignoreファイルの内容を確認し、必要に応じて修正

Windows11特有の問題

問題：文字エンコーディングの問題

```bash
git config --global core.autocrlf true
```

macOS特有の問題

問題：Xcodeコマンドラインツールが必要

```bash
```

```
xcode-select --install
```

セキュリティのベストプラクティス

機密情報の保護

1. **環境変数ファイルの除外**：`.env`ファイルは必ず`.gitignore`に追加
2. **APIキーの分離**：設定ファイルとコードを分離
3. **Personal Access Tokenの管理**：定期的な更新と適切な権限設定

定期的なメンテナンス

1. **不要なブランチの削除**
2. **古いコミットの整理**（必要に応じて）
3. **依存関係の更新**

まとめ

このガイドに従うことで、VSCodeからGitHubへの接続とファイルのプッシュが可能になります。最初は手順が多く感じるかもしれませんが、一度設定してしまえば日常的な操作は非常にシンプルになります。

重要なポイント：

- 初期設定は丁寧に行う
- 意味のあるコミットメッセージを書く
- 定期的にプッシュしてバックアップを取る
- 機密情報は絶対にコミットしない

何か問題が発生した場合は、エラーメッセージを注意深く読み、このガイドのトラブルシューティング部分を参照してください。Git とGitHubの操作に慣れることで、より効率的な開発が可能になります。