

第14回 2026年1月26日（月）1・2限

Python

プログラミング及び演習

Streamlitで可視化アプリ入門



今日のゴール

第13回：アプリ公開と管理

授業の流れ

1

デプロイの仕組み

全体像・用語（Git/GitHub/Cloud）

2

環境準備

Gitインストール・初期設定・VS Code連携

3

GitHub連携

リポジトリ作成・コミット・プッシュ

4

アプリ公開

Streamlit Cloudデプロイ・公開設定

この回の到達目標



デプロイの全体像を理解する

Git / GitHub / Streamlit Cloud の関係性とデータの流れを把握する



実際にアプリをWeb公開する

ローカルにあるアプリをクラウドへ上げ、URLでアクセスできる状態にする



公開後の管理方法を学ぶ

修正の反映手順や、公開範囲の設定など運用に必要な知識を身につける

デプロイの全体像：ローカルからクラウドへ



📌 推奨手順：①Localで動作確認 > ②GitHubへpush > ③Cloudで公開

❗ ※ requirements.txt がないとクラウド上で動きません

用語整理：Git と GitHub（基本）

「自分のPCで管理」するか「クラウドで共有」するか



Git（ローカルPC）

自分のPC内で
「変更履歴」を管理するシステム



バージョン管理

「いつ・誰が・何を」変更したか記録



リポジトリ / ステージング

ファイルの保管庫と一時置き場

主な操作（自分のPC内）

add



commit



GitHub（クラウド）

Gitのデータを
「クラウド上でホスト・共有」する場所



リモートリポジトリ

インターネット上の保管場所



共同開発・共有

複数人で同じコードを編集・管理

主な操作（クラウドへ送信）

Local



push



Remote

i なぜ必要？

「誰が・いつ・何を」変更したか追跡できないと、チーム開発でファイルが先祖返りしたり壊れたりします

Gitの準備とVS Code連携

Windows環境での推奨インストール手順

1



公式サイトからDL

git-scm.com へアクセス

- 1 **Download for Windows** をクリック
(PCのモニター画像の中のボタン)
- 2 "Standalone Installer"の
Git for Windows Setup
(通常は 64-bit) を選択してDL
- 3 インストーラー(exe)を起動

2



エディター設定 (重要)

インストール中の画面操作

画面タイトル:

Choosing the default editor...

プルダウンから選択:

- ☒ Use Visual Studio Code as Git's default editor

! これ以外の設定項目は、すべて初期値のまま
Next 連打でOK

3



インストール確認

ターミナルのメニューからGit Bashを開く

- 1 以下のコマンドを入力

```
$ git --version
```

i 成功例

```
git version 2.43.0...
```

※エラーが出る場合は、ターミナルを再起動してください

⚠ 注意：VS Codeが開いている状態でインストールした場合は、VS Codeの再起動が必要です。

ローカル環境での準備

デプロイ前にPC側で行う設定（1回のみ）

1 Gitの初期設定

VS Code のターミナル（Git Bash）で、名前とメールアドレスを登録します。※これは「誰が変更したか」を記録するために必須です。

▼ 以下のコマンドを1行ずつ実行

```
$ git config --global user.name "Your Name"
$ git config --global user.email "your@email.com"
```

※ "Your Name" や "your@email.com" は自分の情報に書き換えてください


VS Code ターミ
ナル

+


Git
Bash


ポイント


Windowsの方は、VS Codeのターミナル
設定で
「Git Bash」を選択して利用します。


2 ライブラリ情報の準備


Cloud上でアプリを動かすために「どのライブラリが必要か」を伝えるリスト（requirements.txt）を作成し、プロジェクト直下に置きます。

プロジェクトフォルダ構成

 streamlit_app/

 streamlit_ex_01.py (メインのコード)

 data.csv

 requirements.txt ← **これが必須！**

▼ requirements.txt の中身（例）

```
streamlit
pandas
plotly
```

 使用しているライブラリ名だけを列挙します

GitHub連携とpush

リポジトリ作成 → ローカル紐付け → ファイル反映

1



新規リポジトリ作成

- 1 GitHub右上の **+** → **New repository**
- 2 Repository nameを入力 (例: streamlit-app)
- 3 **Public** を選択 (無料プラン用)
- 4 **Create repository** をクリック

🔗 URLを控える

`https://github.com/user/repo.git`

2



ローカルで紐付け

VS Codeのターミナル (Git Bash) で実行

```
# Git管理を開始
git init

# リモートURLを設定
git remote add origin <URL>

# 設定を確認
git remote -v
```

📌 ポイント

<URL> の部分は Step 1 で控えたURLを貼り付けます。

3



反映 (Push)

変更を記録してクラウドへ送信

```
# 全ファイルをステージング
git add .

# 記録 (コミット)
git commit -m "1st commit"

# GitHubへ送信
git push origin master
```

✓ GitHub上でファイルを確認できれば成功

➤ コマンド操作は「正確さ」が重要です。エラーが出たらスペルやスペースを確認しましょう。

Streamlit Cloud でデプロイ

GitHub上のコードを読み込んでWebアプリとして公開

1



連携 & 開始

- 1 Streamlit Cloudへサインイン
(GitHubアカウントを使用)
- 2 右上の **New app** をクリック
- 3 **Use existing repo** を選択



GitHub連携が必須
初回のみ許可設定が必要です

2



デプロイ設定

公開する対象を指定します

Repository **your-repo-name**

Branch **main**

Main file **streamlit_ex_01.py**

Advanced settingsからPythonバージョン
(3.12)やSecrets設定が可能

Deploy!

3



公開・共有

- 1 成功するとアプリが起動
URLが発行されます

Share設定

Public

誰でも閲覧可能 (デフォルト)

Specific people

メールアドレスで招待



デプロイ完了後も、GitHubのmainブランチを更新 (push) するだけでアプリは自動更新されます。