

AIエージェント入門

#1 AIエージェントとは / ツールの基本

2025/10/2 Shinji Akematsu

自己紹介

- 明松 真司（あけまつ しんじ）
- 株式会社PolarTech 代表取締役 社長
- 釧路高専情報工学科 卒業
- 東北大学理学部数学科 卒業
- 滋慶学園COMグループ 名誉教育顧問
- 宮城県情報サービス産業協会MISA 講師
- 円ポイント株式会社 サポート
- ex. スキルアップAI株式会社／尚絅学院大学／釧路工業高等専門学校／ポリテクセンター宮城／仙台応用情報学研究振興財団 etc
- 高専のための学習塾「ナレッジスター」創業
- 【著書】
 - 「線形空間論入門」(プレアデス出版)
 - 「徹底攻略ディープラーニングG検定問題集第1版」(インプレス)
 - 「徹底攻略ディープラーニングG検定問題集第2版」(インプレス)
 - 「Pythonで超らくらくに数学をこなす本」(オーム社)
 - 「1週間でLaTeXの基礎が学べる本」(インプレス)
 - 「基礎数学」(滋慶出版) 「AI基礎」(滋慶出版)
 - 「1週間でブロックチェーンの基礎が学べる本」(インプレス)

AIのいままでとこれから

第4次AIブーム（2020年～） 生成AI

- 第4次AIブームは2020年代に訪れた
- **生成AI（何かを生み出せるAI）** が世界的に急激に流行
- Transformerの登場を契機に、GPTをはじめとする大規模言語モデル（LLM）が爆発的に発展
- 現在は画像生成AI、音楽生成AI、動画生成AIなど、さまざまな生成AIが急激に発展している

弱いAI、強いAI

- **弱いAI（特化型AI）** は、特定のタスクに特化したAI（例：翻訳、画像認識など）
- **強いAI（汎用AI）** は、人間のように自律的に思考・判断できるAI（まだ実現していない）
- 現在使われているAIはすべて「弱いAI」で、身近な技術はこの範囲内。



弱いAI



強いAI

シンギュラリティ（技術的特異点）

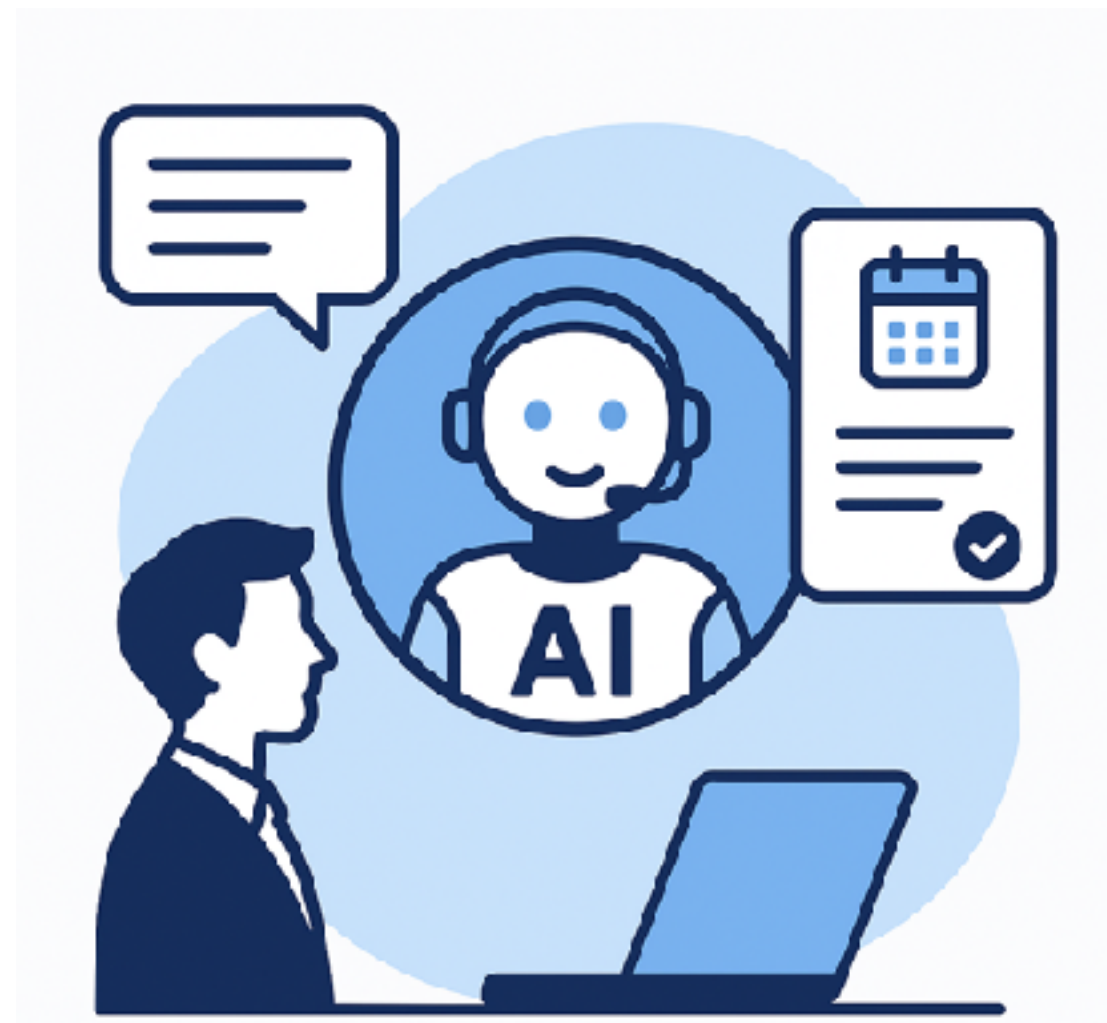
- シンギュラリティ（技術的特異点）とは、AIが人間の知能を超える転換点のこと
- 多くの専門家が**2045年頃**に到来すると予測（※諸説あり）
- その後はAIがさらにAIを作り出すなど爆発的な技術進化が起こるとされる



AIエージェントの登場

AIエージェント

- ユーザーの指示に基づいて、タスクを自律的に完了させるAI（例：予定調整、情報収集、提案作成など）。
- 自然言語でやりとりしながら、ツールやサービスを操作して処理を進める。
- 単なる回答ではなく、「～して、次に～して…」のような一連の行動を実行できる。
- ユーザーの履歴や好みに応じて振る舞いを変えるパーソナライズ能力を持つ。





Devin

CLAUDE

CODE

Gemini - CLI



```

1  // This is the main function of the program.
2  // It takes a string as input and returns a boolean value.
3  // The function is named 'isPalindrome' and is located in the 'main' function.
4  // The function is defined as follows:
5  // bool isPalindrome(string s)
6  // {
7  //     // Convert the string to lowercase.
8  //     string lower = s;
9  //     transform(lower.begin(), lower.end(), lower.begin(), ::tolower);
10    // Remove spaces and punctuation.
11    string cleaned;
12    for (char c : lower) {
13        if (isalnum(c)) {
14            cleaned += c;
15        }
16    }
17    // Check if the string is a palindrome.
18    int left = 0;
19    int right = cleaned.length() - 1;
20    while (left < right) {
21        if (cleaned[left] != cleaned[right]) {
22            return false;
23        }
24        left++;
25        right--;
26    }
27    return true;
28 }
29
30 // This is the main function of the program.
31 // It takes a string as input and returns a boolean value.
32 // The function is named 'isPalindrome' and is located in the 'main' function.
33 // The function is defined as follows:
34 // bool isPalindrome(string s)
35 // {
36 //     // Convert the string to lowercase.
37 //     string lower = s;
38 //     transform(lower.begin(), lower.end(), lower.begin(), ::tolower);
39 //     Remove spaces and punctuation.
40 //     string cleaned;
41 //     for (char c : lower) {
42 //         if (isalnum(c)) {
43 //             cleaned += c;
44 //         }
45 //     }
46 //     Check if the string is a palindrome.
47 //     int left = 0;
48 //     int right = cleaned.length() - 1;
49 //     while (left < right) {
50 //         if (cleaned[left] != cleaned[right]) {
51 //             return false;
52 //         }
53 //         left++;
54 //         right--;
55 //     }
56 //     return true;
57 // }
58
59 // This is the main function of the program.
60 // It takes a string as input and returns a boolean value.
61 // The function is named 'isPalindrome' and is located in the 'main' function.
62 // The function is defined as follows:
63 // bool isPalindrome(string s)
64 // {
65 //     // Convert the string to lowercase.
66 //     string lower = s;
67 //     transform(lower.begin(), lower.end(), lower.begin(), ::tolower);
68 //     Remove spaces and punctuation.
69 //     string cleaned;
70 //     for (char c : lower) {
71 //         if (isalnum(c)) {
72 //             cleaned += c;
73 //         }
74 //     }
75 //     Check if the string is a palindrome.
76 //     int left = 0;
77 //     int right = cleaned.length() - 1;
78 //     while (left < right) {
79 //         if (cleaned[left] != cleaned[right]) {
80 //             return false;
81 //         }
82 //         left++;
83 //         right--;
84 //     }
85 //     return true;
86 // }
87
88 // This is the main function of the program.
89 // It takes a string as input and returns a boolean value.
90 // The function is named 'isPalindrome' and is located in the 'main' function.
91 // The function is defined as follows:
92 // bool isPalindrome(string s)
93 // {
94 //     // Convert the string to lowercase.
95 //     string lower = s;
96 //     transform(lower.begin(), lower.end(), lower.begin(), ::tolower);
97 //     Remove spaces and punctuation.
98 //     string cleaned;
99 //     for (char c : lower) {
100 //         if (isalnum(c)) {
101 //             cleaned += c;
102 //         }
103 //     }
104 //     Check if the string is a palindrome.
105 //     int left = 0;
106 //     int right = cleaned.length() - 1;
107 //     while (left < right) {
108 //         if (cleaned[left] != cleaned[right]) {
109 //             return false;
110 //         }
111 //         left++;
112 //         right--;
113 //     }
114 //     return true;
115 // }

```

Gemini CLIがすごい

- **GeminiCLI**はGoogle社によるAIエージェント
- かなりの高機能を持ちながら、「ほぼ無料」でつかえる
- ※ Gemini pro はリミットあり。リミットを超えたらGemini flashに切り替わる

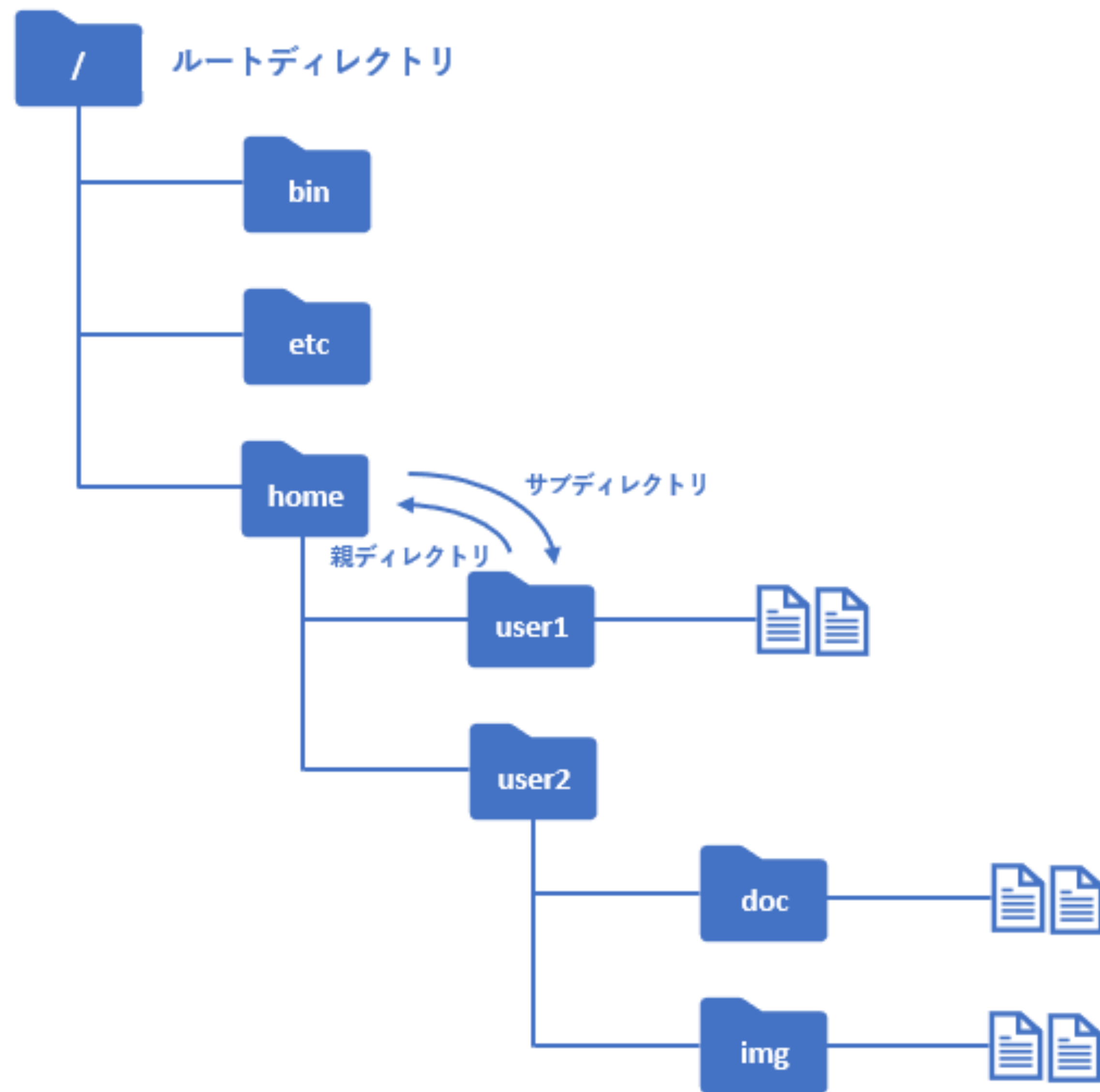


実演

ターミナルの操作

ターミナル

- コンピュータに直接命令を送るための文字ベースのインターフェース。
(**Command Line Interface**)
- マウス操作ではなく、キーボード入力（コマンド）で処理を実行する。
- GUI（Graphical User Interface）ではできない細かい制御や自動化を行えるため、開発者・エンジニアに必須のツール。
- Linux、macOS、Windows（**PowerShell**やコマンドプロンプト）など、各OSに標準的に備わっている。
- **Gemini CLIは、ターミナルで動く！**



ツリー構造

とりあえず覚えてほしいコマンド①

- **ls** (**list**)
- **cd** (**change directory**)

とりあえず覚えてほしいコマンド②

- **mkdir** (make directory)
- **cat** (concatenate)
- **pwd** (print working directory)

実演

練習

- Slackで共有した「Japan_dir.zip」を解凍し、適当なところに配置してください。
- **1. Japanディレクトリに移動**
Japanディレクトリに移動してください。
- **2. 東京の北区へ移動**
Tokyo/Kita に移動してください。
- **3. 赤羽に行ってみる**
Akabane に移動して、そこにあるファイルの名前を調べてください。
- **4. ラーメン屋のメニュー確認**
赤羽にある「MenyaRyu.txt」を開いて、どんな飲み物があるか答えてください。
- **5. 大阪の梅田へ移動**
Osaka/Kita/Umeda に移動してください。そこにはどんなお店がありますか？
- **6. ホルモン太郎の料理**
Osaka/Namba/Arcade にある「HorumonTaro.txt」を開いて、肉料理を2つ探してください。

Git

Git

- ソースコードやファイルの 変更履歴を管理するバージョン管理システム。
- いつ、誰が、どのような変更を行ったかを記録し、過去の状態に戻すことができる。
- チームでの共同開発を効率化
- リモートリポジトリ（例：GitHub, GitLab, Bitbucket）と連携して、世界中どこからでも協働できる。
- ソフトウェア開発だけでなく、ドキュメントやデータ分析プロジェクトなど幅広い分野で利用されている。



Github

- Gitで管理したファイルをインターネット上で共有・保存できるサービス。
- 開発者が作ったコードを公開・非公開で管理できる。
- リモートリポジトリとして利用し、複数人での共同開発を可能にする。
- 世界最大のコード共有プラットフォームであり、オープンソースの中心的存在。



リモートリポジトリ



ローカルリポジトリ



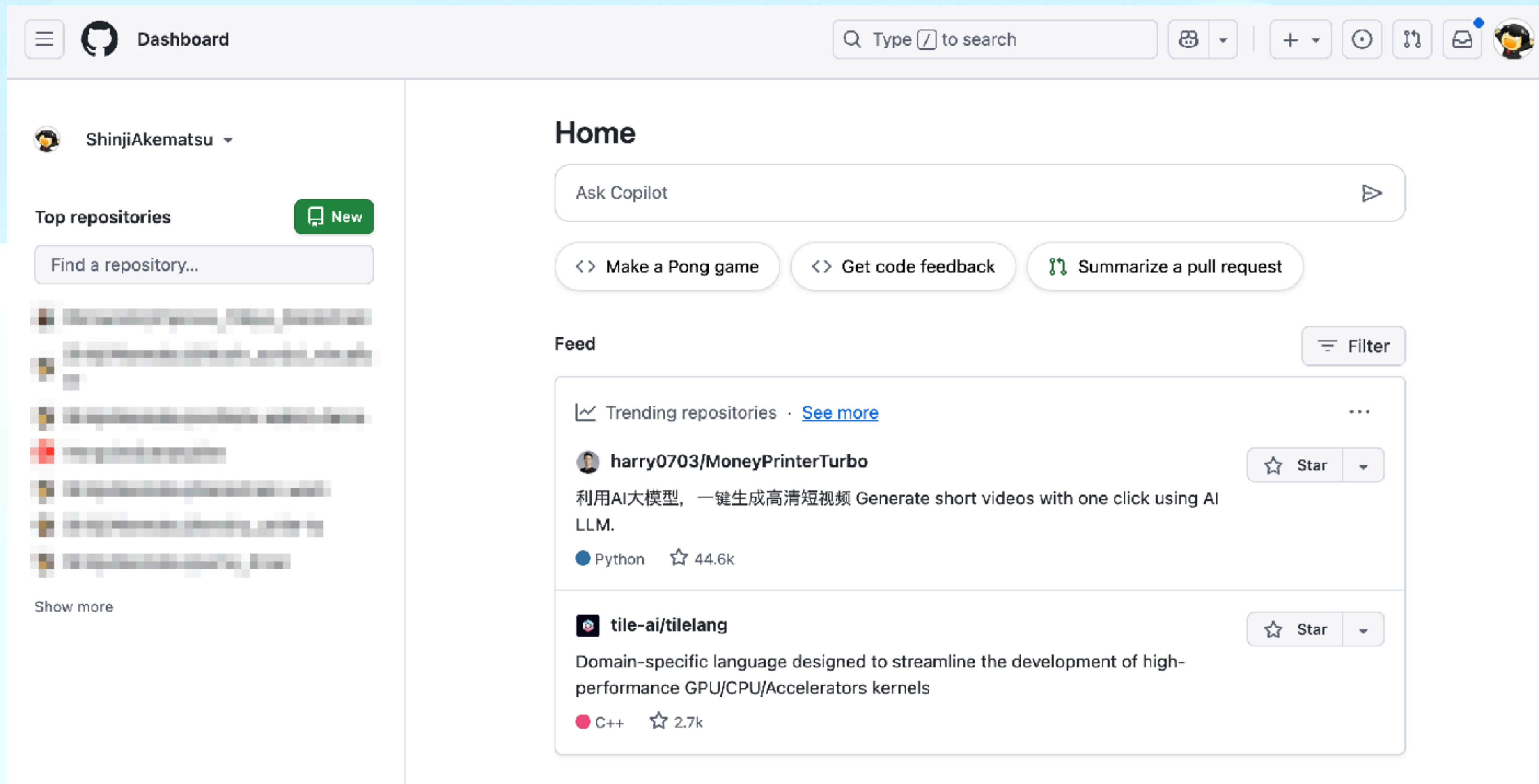
ローカルリポジトリ



ローカルリポジトリ

① Githubにログイン

- まずはGithubにログインしましょう。



② Githubにログイン

- 右上の「+」から New repository を選ぶ
- リポジトリ名を決めて「Create repository」をクリック
- 画面に出てきたURL（HTTPS か SSH）をコピー

③ 手元フォルダをGitフォルダに

- 以下のように、現在のディレクトリをgit管理スタート

```
> cd プロジェクトのフォルダ
```

```
> git init
```

④ リモートリポジトリと接続

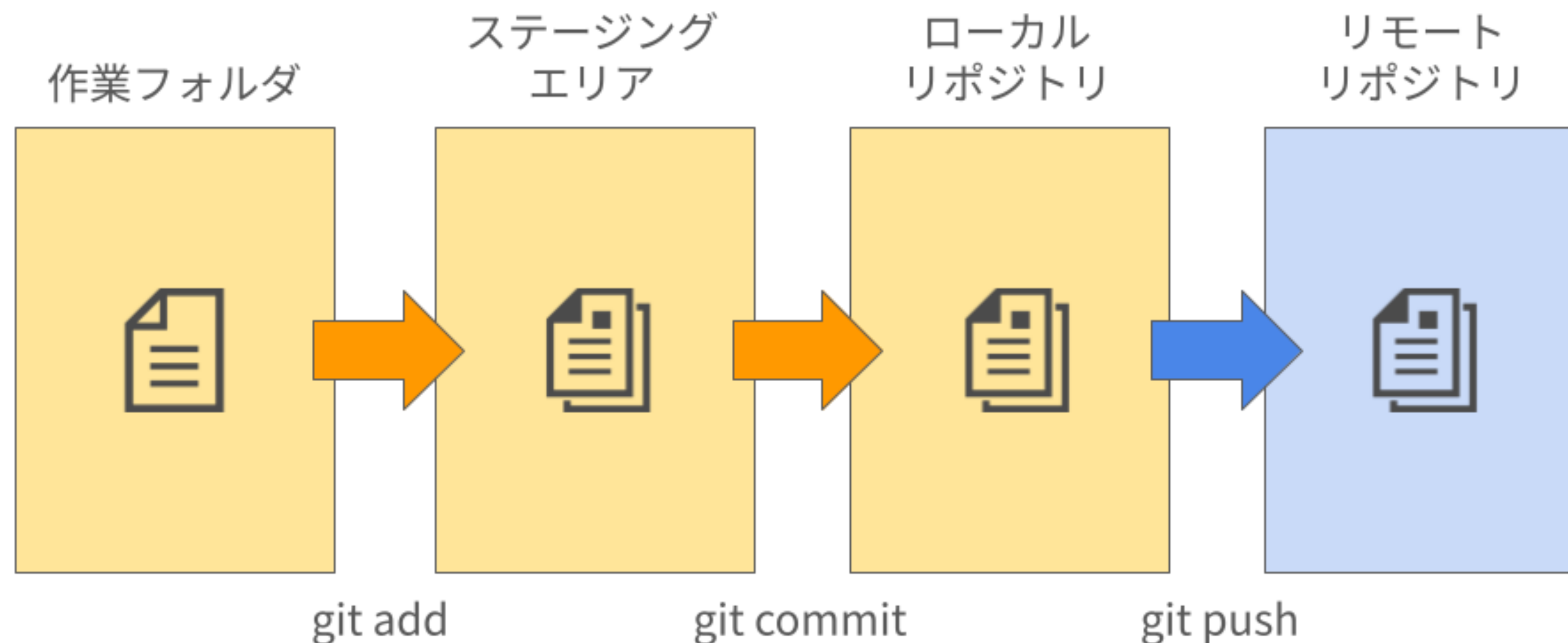
- 以下のように、リモートリポジトリと現在のディレクトリを接続する

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


⑤ ファイルを登録して最初のコミット

- add → commit

```
> git add .  
> git commit -m "最初のコミット"
```

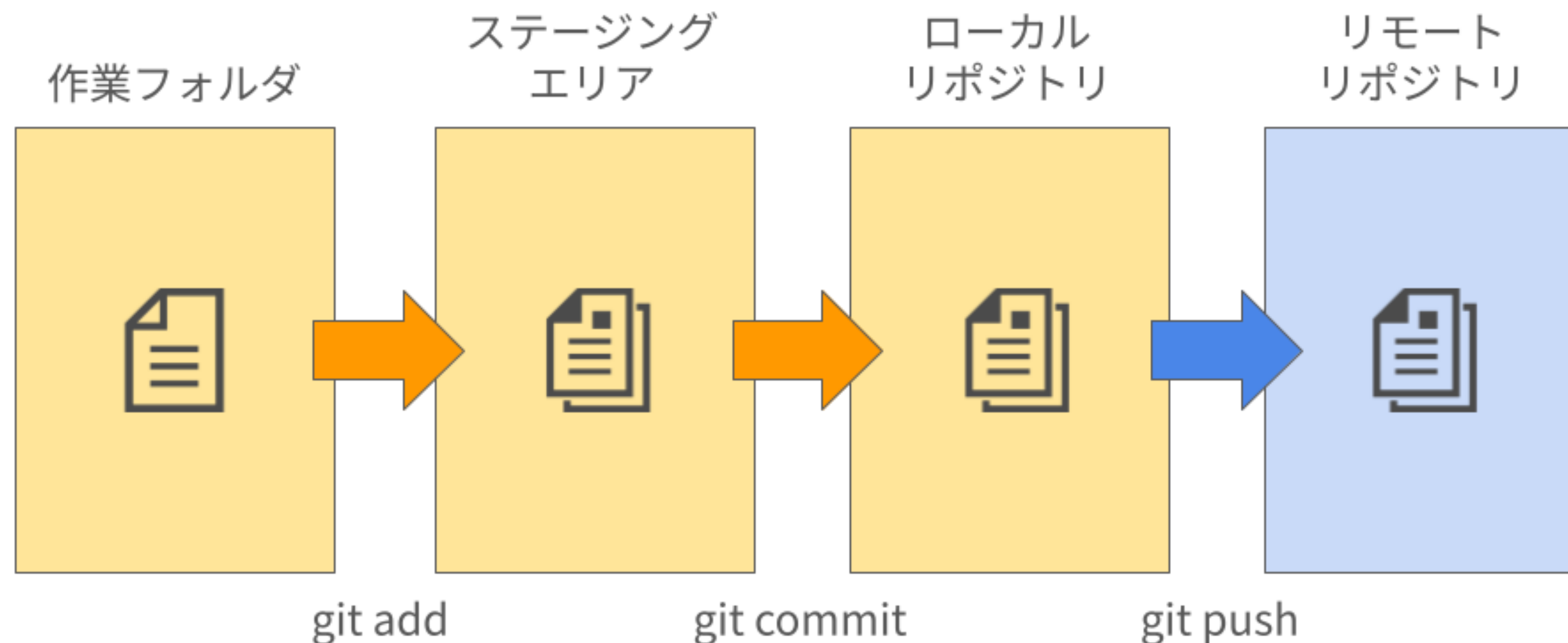


⑥ Githubにpushする

- ブランチを設定してpush

```
> git branch -M main
```

```
> git push -u origin main
```



以降のステップ

1. ファイルを変更したら

```
bash
```

```
git add .
```

2. コメントを付けて保存（コミット）

```
bash
```

```
git commit -m "ここに変更内容を書く"
```

3. GitHubへ反映

```
bash
```

```
git push
```

練習

- 手順書に従って、Gitを使う練習をしてみましょう。

Git/GitHub 練習問題 手順書

ゴール

GitHubとローカルをつなげて、add → commit → pushの流れを体験する。

① GitHubでリポジトリを作成

1. ブラウザでGitHubにログイン
2. 「New repository」からリポジトリを作成
 - 名前: practice-git
 - READMEは作らないでOK
3. 作成後に表示されるURL (HTTPS) をコピー

② ローカルで練習用フォルダを作る

```
mkdir practice-git  
cd practice-git
```

③ テキストファイルを作る

```
echo "はじめてのGit練習です。" > hello.txt
```


練習

- 手順書に従って、Gitを使う練習をしてみましょう。

Git/GitHub 練習問題 手順書

ゴール

GitHubとローカルをつなげて、add → commit → pushの流れを体験する。

① GitHubでリポジトリを作成

1. ブラウザでGitHubにログイン
2. 「New repository」からリポジトリを作成
 - 名前: practice-git
 - READMEは作らないでOK
3. 作成後に表示されるURL (HTTPS) をコピー

② ローカルで練習用フォルダを作る

```
mkdir practice-git  
cd practice-git
```

③ テキストファイルを作る

```
echo "はじめてのGit練習です。" > hello.txt
```