

# AI とペアプログラミング (Aider、Ollama)

## 目次

はじめに .....	1
1. 上流工程で AI ができること .....	1
2. システム開発の将来像 (GitHub Copilot Workspace と Aider) .....	2
2.1. Aider の操作 .....	2
2.2. チャットモード .....	3
2.3. 新規ファイルの作成 .....	3
2.4. 機能を指定してクラスを生成し機能追加 .....	5
2.5. プロジェクトの一括修正 .....	7
2.6. CSV ファイルのバッチ処理クラスを生成し Aider が補正 .....	8
2.7. フレームワークを使った WEB アプリの生成とブラッシュアップ .....	9
2.8. (日本語でチャット) Spring batch を使った pdf 出力アプリを生成 .....	12
2.8.1. (Aider に質問) 作ったアプリの実行方法を聞いてみる .....	15
3. Aider のインストール/環境作成 .....	16
3.1. Python 3 .....	16
3.2. Git .....	16
3.3. Ollama と Llama 3.1 .....	16
3.4. Aider 動作環境の作成 .....	18
3.5. Aider の起動 .....	18

# AI とペアプログラミング (Aider、Ollama)

はじめに

2022 年 11 月に ChatGPT が公開されて大規模言語モデル (LLM) の可能性が認知されると色々な分野に応用され、システム開発の分野では LLM を使ったコード補完が当たり前のツールになろうとしています。現状ではマシン性能が壁になっていますが、以下の事情が導入を後押ししています。

- ① 企業では多数のシステム開発でいまだに Java が使われていますが<sup>1</sup>、Java は仮想マシン (JVM) の強化ではなく言語仕様 (コンパイラによる操作) だけ他のオブジェクト指向言語に追随させようとしているため、JDK1.5 以降のプリミティブ型 (int 他) とラッピングクラス (Integer 他) の中途半端な相互変換や定数のインライン化等で潜在的なバグが作られやすく、コードオブジェクトモドキの関数型インタフェースも意味不明のコンパイルエラーで初学者を悩ませます。
- ② LLM を通して世界中の知恵や経験を再利用でき、生産性と信頼性の向上が期待できます。また、高機能なクラスやメソッドはコード例なくして使うことができません。

そして ChatGPT の公開から 1 年半、開発ツールの目標はコードの生成からより上流の要件定義や設計工程へ移ってきています。

## 1. 上流工程で AI ができること

いくつかの論文は AI/LLM を使って次のことができると予測し、いくつかは実現しています。

- ・要件の抽出と分析
- ・自動コード生成によるプロジェクトのプロトタイピング
- ・コードパターン、データフロー、エラーログを分析しソフトウェアの問題を自動的に特定
- ・コードを分析し説明文を生成する
- ・コードのリファクタリング (アンチパターンの指摘、可読性の向上、他)
- ・プロジェクトのタイムライン、リソース要件、潜在的なリスクと統計データから結果を予測
- ・コードを解析して脆弱性やボトルネックを見つけたり、テストシナリオをシミュレートする

仕様の不完全な部分を指摘して改善案を提示したり GUI を半自動生成する AI ツールは既に出現しています。今は簡単なゲームが作れる程度の機能ですが、使い続けるうちに社内で使っている言葉の意味 (例えば、リードタイムや最低在庫数、値引きや利用ポイント) やルール (例えば、納入/出庫倉庫の優先順等々) をローカルキャッシュに蓄積して適用できるようになると推測できます。

オープンソースの複数の LLM で試したところ日本語の文書も十分に意思疎通が可能な表現に要約できます。利用者が書いた自然文の要求事項を整理し矛盾点を問合せ仕様化しプログラム生成ができるようになる道筋が見えてきていますが、LLM は人間の過去の試行錯誤の蓄積に過ぎず新しいものを生み出す欲望も持っていないので AI が「優秀な助手」の域を超えることは今の延長線上にはありません。見通せる将来に必要な技能は AI から最適な情報を引き出す AI プロンプタの技術で、Web 検索で欲しい情報を探し出すのと似て的確なキーワードの選択と組合せの習熟が重要になります。

プロンプト自体もマクロ化が進めば単純なプログラムはコードを必要とせず、システム運用でも個別のコマンドではなく利用者の目的を文書で伝えれば事足りるノーコードの世界が広がります。

---

<sup>1</sup> 独立行政法人情報処理推進機構 (IPA)- ソフトウェア開発 分析データ集 2022

<https://www.ipa.go.jp/digital/chousa/metrics/hjuojm000000c6it-att/000102171.pdf>

## AI とペアプログラミング (Aider、Ollama)

### 2. システム開発の将来像 (GitHub Copilot Workspace と Aider)

AI 利用による上流工程開発ツールは GitHub Copilot Workspace が既に商用化していますが、代替えを狙うオープンソースプロジェクトが幾つかあり、そのうちの一つに Aider があります。

GitHub Copilot Workspace はエディターを一体化した開発環境で、謳い文句は以下のとおりです。  
『Copilot Workspace では、開発者は自然言語でブレインストーミング、計画、構築、テスト、コード実行を行うことができます。』

Aider の目標も Copilot Workspace と主要部分が重なります。特徴はコマンドラインを通して LLM と会話しながら開発を行うという軽量な操作感です。守備範囲は少し異なりますが開発者との会話やコードの生成はどちらも LLM に依存しているので、AI を利用したシステム開発の将来像がどのようになるのか Aider の操作から雰囲気を感じることができると思います。

#### 2.1. Aider の操作

PowerShell のターミナルから Aider を起動すると実行環境の情報と、コマンドやチャットを受け付けるプロンプトが表示されるので、このプロンプトを通して Aider を操作します。

下のイメージは、LLM は Ollama を通して Llama 3.1 を使用し Git は不使用の環境で起動したものです。一番下に出ている > の部分に、Aider に対するコマンドやチャット内容を入力します。使用可能なコマンドは 1 桁目に / を入力すると候補が表示されます。

```
(aider) PS C:\Users\remoteuser> aider --no-git --model ollama/llama3.1  
Aider v0.56.0  
Model: ollama/llama3.1 with whole edit format  
Git repo: none  
Repo-map: disabled  
Use /help <question> for help, run "aider --help" to see cmd line args  
> |
```

## AI とペアプログラミング (Aider、Ollama)

### 2.2. チャットモード

Aider はチャットをしながら作業を進めます。チャットの目的により以下の 3 モードがあります。

- code : リクエストに従いコードに変更を加えます (起動時のデフォルト)
- ask : コードに関する質問に回答し、コードを編集することはありません
- help : Aider の使用方法、設定、トラブルシューティングなどに関する質問に回答します。

/chat-mode モード で切り替えます

※/chat-mode help で日本語の質問には日本語で回答されます

```
> /chat-mode help
Aider v0.56.0
Model: ollama/llama3.1 with help edit format
Git repo: none
Repo-map: disabled

help> javaの開発を始めるにはどうすればいい？

Assistant:
まず、Aider CLIツールをインストールしてください。次に、以下のコマンドを実行して、Javaプロジェクトを作成します。

aider init java

これで、基本的なJavaプロジェクトが作成されます。
また、Aiderのドキュメントを参照することもおすすめします。
• https://aider.chat/docs/usage.html
• https://aider.chat/docs/java.html
これらのドキュメントでは、Aider CLIツールの使い方や、Java開発に使用できる機能について詳しく説明しています。

Tokens: 337 sent, 168 received.

help>
```

※ 上の Aider の回答にある `aider init java` は Git のプロジェクトを作るためのコマンドなので、既に Git のリポジトリに存在する場合や使わない場合は実行不要です

### 2.3. 新規ファイルの作成

以下のコマンドで作業ディレクトリ内にコマンドで指定したファイルが作られます。

`aider ファイル名`

新規ファイルには aider が簡単な main メソッドを書くので、これを編集 (Aider に修正指示) して目的のソースを作っていきます。但し、新規クラスを作らなくてもチャットで機能要件を伝えると Aider が自動でファイル名/クラス名を仮決めして作ってくれるので、作られたクラスの名前を後で変えた方が手間が少ないかもしれません。

## AI とペアプログラミング (Aider, Ollama)

白抜き部分が Aider が初期生成したコード

```
(aider) PS C:\Users\remoteuser> aider --no-git --model ollama/llama3.1

Aider v0.56.0
Model: ollama/llama3.1 with whole edit format
Git repo: none
Repo-map: disabled
Use /help <question> for help, run "aider --help" to see cmd line args

> aider sample.java

sample.java

// sample.java
import java.util.Scanner;

public class Sample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Hey, what's your name?");
        String name = scanner.nextLine();
        System.out.println("Hey " + name + ", nice to meet you!");
    }
}

I made the greeting more casual by changing it from a formal "Hello" to an informal "Hey". I also added a simple program that asks for the user's name and then greets them with the new, casual greeting.

Tokens: 560 sent, 123 received.

sample.java
Create new file? (Y)es/(N)o [Yes]: |
```

ファイル保存の確認

```
sample.java
Create new file? (Y)es/(N)o [Yes]:
Applied edit to sample.java

sample.java
> |
```

Y または空 enter でターミナルに表示された内容が  
作業ディレクトリに書き出され、編集が始まります。

```
sample.java
> 内容を表示してください

sample.java

@@ -1,4 +1,5 @@
// sample.java
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Hey, what's your name?");
        String name = scanner.nextLine();
-       System.out.println("Hey " + name + ", nice to meet you!");
+       System.out.println("Hey " + name + ", it was nice meeting you!");
    }
}

I changed the last line of the greeting to be more casual by changing it from "nice to meet you!" to "it was nice meeting you!".

Tokens: 834 sent, 108 received.
Applied edit to sample.java

sample.java
> |
```

## AI とペアプログラミング (Aider、Ollama)

### 2.4. 機能を指定してクラスを生成し機能追加

【チャット指示】素数を計算するクラスを作ってください…敬語でなくても大丈夫 (だと思う)

```
sample.java
> 素数を計算するクラスを作ってください

I'll create a class to calculate prime numbers.

First, I need to understand what changes are needed. You want me to add a class that calculates prime numbers. Is that correct?

If so, I will:

1 Determine if any code changes are needed.
2 Explain any needed changes.
3 If changes are needed, output a copy of each file that needs changes.

Let's get started!

I'll create a new file called PrimeCalculator.java. Here it is:

PrimeCalculator.java

// PrimeCalculator.java
public class PrimeCalculator {
    public static boolean isPrime(int number) {
        if (number <= 1) {
            return false;
        }
        for (int i = 2; i * i <= number; i++) {
            if (number % i == 0) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        System.out.println(isPrime(25)); // false
        System.out.println(isPrime(23)); // true
    }
}

This class has a method isPrime that checks if a given number is prime. It returns true if the number is prime and false otherwise.

Please review this code and let me know if it meets your requirements!

Tokens: 979 sent, 269 received.

PrimeCalculator.java
Create new file? (Y)es/(N)o [Yes]: |
```

※ この場合、PrimeCalculator クラスとその中に isPrime メソッド、main メソッドが作られました

## AI とペアプログラミング (Aider、Ollama)

【機能追加の指示】 isPrime のパラメータを 1 から 100 まで変えて呼び出す処理を main メソッドに追加してください。

```
PrimeCalculator.java sample.java
> isPrimeのパラメータを1から100まで変えて呼び出す処理をmainメソッドに追加してください。

I'll add the requested functionality to the PrimeCalculator class.

First, I need to understand what changes are needed. You want me to modify the isPrime method to take a parameter from 1 to 100 and also add a process to call this method with different parameters in the main method. Is that correct?

If so, I will:

1 Determine if any code changes are needed.
2 Explain any needed changes.
3 If changes are needed, output a copy of each file that needs changes.

Here's the updated PrimeCalculator.java file:

PrimeCalculator.java

@@ -1,4 +1,5 @@
// PrimeCalculator.java

    public static void main(String[] args) {
        System.out.println(isPrime(25)); // false
        System.out.println(isPrime(23)); // true
+       for (int i = 1; i <= 100; i++) {
+           if (isPrime(i)) {
+               System.out.println("Number " + i + " is prime.");
+           }
+       }
    }
}

I added a for loop in the main method to call the isPrime method with numbers from 1 to 100 and print out which numbers are prime.

Please review this code and let me know if it meets your requirements!

Tokens: 1.4k sent, 337 received.
Applied edit to PrimeCalculator.java

PrimeCalculator.java sample.java
>
```

できたアプリを実行した結果 (java 11 以降であれば、コンパイル[javac]せず実行できます)

```
> java PrimeCalculator.java
false
true
Number 2 is prime.
Number 3 is prime.
Number 5 is prime.
Number 7 is prime.
Number 11 is prime.
Number 13 is prime.
Number 17 is prime.
Number 19 is prime.
Number 23 is prime.
Number 29 is prime.
Number 31 is prime.
Number 37 is prime.
Number 41 is prime.
Number 43 is prime.
Number 47 is prime.
Number 53 is prime.
Number 59 is prime.
Number 61 is prime.
Number 67 is prime.
Number 71 is prime.
Number 73 is prime.
Number 79 is prime.
Number 83 is prime.
Number 89 is prime.
Number 97 is prime.
```

## AI とペアプログラミング (Aider、Ollama)

### 2.5. プロジェクトの一括修正

作業ディレクトリに入っている資料を一括して修正することができます。対象はプロンプトの上に表示されている (下例の場合、PrimeCalculator.java と sample.java) です。

```
PrimeCalculator.java sample.java
> 全ての println を print に変えてください

sample.java

@@ -1,4 +1,5 @@
// sample.java

public class Sample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
-       System.out.println("Hey, what's your name?");
+       System.out.print("Hey, what's your name?");
        String name = scanner.nextLine();
-       System.out.println("Hey " + name + ", it was nice meeting you!");
+       System.out.print("Hey " + name + ", it was nice meeting you!");
    }
}

PrimeCalculator.java

@@ -1,4 +1,5 @@
// PrimeCalculator.java
return true;
}

public static void main(String[] args) {
-   System.out.println(isPrime(25)); // false
-   System.out.println(isPrime(23)); // true
+   System.out.print(isPrime(25)); // false
+   System.out.print(isPrime(23)); // true
    for (int i = 1; i <= 100; i++) {
        if (isPrime(i)) {
-           System.out.println("Number " + i + " is prime.");
+           System.out.print("Number " + i + " is prime.");
        }
    }
}

Tokens: 1.4k sent, 243 received.
Applied edit to PrimeCalculator.java
Applied edit to sample.java

PrimeCalculator.java sample.java
> |
```

## AI とペアプログラミング (Aider、Ollama)

### 2.6. CSV ファイルのバッチ処理クラスを生成し Aider が補正

【指示】 csv ファイルを読んで一覧表を作る Java のクラスを作ってください

```

> csvファイルを読んで一覧表を作るJavaのクラスを作ってください

System:
Act as an expert software developer.
Take requests for changes to the supplied code.
If the request is ambiguous, ask questions.
Always reply to the user in the same language they are using.
Once you understand the request you MUST:
  1 Determine if any code changes are needed.
  2 Explain any needed changes.
  3 If changes are needed, output a copy of each file that needs changes.

User:
csvファイルを読んで一覧表を作るJavaのクラスを作ってください

Assistant:
Ok, I will create a class to read CSV files and generate a table in Java.
I'll assume it's a simple text-based table for now. If you need something more complex later, we can revisit the design.
TableGenerator.java

// entire file content ...
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class TableGenerator {
  (中略)

```

不足している情報・条件は Aider/LLM が適宜補足しソースの初期生成を行った旨のメッセージ

```

I've updated the code to include error handling for file reading and improved the generateTable method to print a more visually appealing table.
TableGenerator.java

// entire file content ...
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class TableGenerator {
  public List<String[]> readCsv(String filename) throws IOException {
    List<String[]> data = new ArrayList<>();

```

生成に続けて Aider がソースチェックを行いエラー処理の追加とテーブルの表示改善に対する更新を実行した旨のメッセージ

(続く)

## AI とペアプログラミング (Aider、Ollama)

(続き)

```
Main.java
// entire file content ...
import java.io.IOException;

public class Main {
    public static void main(String[] args) throws IOException {
        TableGenerator generator = new TableGenerator();
        String filename = "example.csv";
        List<String[]> data = generator.readCsv(filename);
        if (data != null) {
            generator.generateTable(data);
        }
    }
}

Tokens: 1.2k sent, 913 received.
TableGenerator.java
Create new file? (Y)es/(N)o [Yes]:
Main.java
Create new file? (Y)es/(N)o [Yes]:
Applied edit to TableGenerator.java
Applied edit to Main.java
Main.java TableGenerator.java
> |
```

### 2.7. フレームワークを使った WEB アプリの生成とブラッシュアップ

**【指示】** Spring を使って、csv ファイルを表示する WEB サービスを作ってください

```
> Springを使って、csvファイルを表示するWEBサービスを作ってください

I will create a Spring-based web service to display CSV files.

First, I need to determine the requirements. You want a web service that can display CSV files using Spring. This implies that we'll need to handle file uploads and parsing of CSV data.

To proceed, I have a few questions:

1 Do you want to store the uploaded CSV files in a database or simply process them on-the-fly?
2 What kind of error handling do you want for cases where the CSV file is malformed or cannot be parsed?
3 Would you like to add any additional features, such as filtering or sorting data within the CSV file?

Please answer these questions so I can better understand your requirements.

(And don't worry, I'll use the correct file listing format when suggesting changes to the files.)

Tokens: 1.1k sent, 160 received.
```

[Aider の回答…2 行目～8 行目和訳]

まず、要件を決定する必要があります。Spring を使用して CSV ファイルを表示できる Web サービスが必要で  
す。これはつまり、ファイルのアップロードと CSV データの解析を処理する必要があります。

次に進むにあたり、いくつか質問があります。

- 1 アップロードされた CSV ファイルをデータベースに保存しますか、それとも単にその場で処理しますか？
- 2 CSV ファイルの形式が正しくない場合、または解析できない場合、どのようなエラー処理が必要ですか？
- 3 CSV ファイル内のデータのフィルタリングや並べ替えなどの追加機能を追加しますか？

お客様の要件をより深く理解するために、これらの質問に答えてください。

## AI とペアプログラミング (Aider、Ollama)

【指示の訂正と追加】サーバ上の csv ファイルを表示する WEB アプリを作ってください。表示する CSV は画面上で並び替えができるようにしてください。

```
> サーバ上の csv ファイルを表示する WEB アプリを作ってください。表示する CSV は画面上で並び替えができるようにしてください。
```

```
I will create a Spring-based web service to display CSV files.
```

```
First, I'll create a new file CsvDisplayController.java to handle the CSV file display functionality. Here's the initial content:
```

```
CsvDisplayController.java
```

```
// entire file content ...  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.web.bind.annotation.GetMapping;  
  
@Controller  
public class CsvDisplayController {  
    @GetMapping("/csv")  
    public String displayCsv(Model model) {  
        // TO DO: implement CSV file display logic here  
        return "csv";  
    }  
}
```

```
Next, I'll create a new file csv.html to display the CSV data. Here's the initial content:
```

```
csv.html
```

```
// entire file content ...  
<!DOCTYPE html>  
<html xmlns:th="http://www.thymeleaf.org">  
<head>  
    <title>Csv Display</title>  
</head>  
<body>  
    <h1>Csv Display</h1>  
    <!-- TO DO: implement CSV data display logic here -->  
</body>  
</html>
```

```
Now, I need to determine how to handle the CSV file display logic. You want to be able to display a CSV file that is stored on the server and allow users to sort the data within the table.
```

```
To proceed, I have a few questions:
```

- 1 Do you want to use a library like OpenCSV to parse the CSV file, or do you want to implement your own parsing logic?
- 2 How would you like to handle sorting of the CSV data? Would you like to add a dropdown menu with sorting options (e.g., ascending/descending), or would you prefer a more complex sorting mechanism?

```
Please answer these questions so I can better understand your requirements.
```

```
(And don't worry, I'll use the correct file listing format when suggesting changes to the files.)
```

```
Tokens: 1.4k sent, 374 received.
```

```
CsvDisplayController.java
```

```
Create new file? (Y)es/(N)o [Yes]:
```

## AI とペアプログラミング (Aider、Ollama)

【指示】 ソートはブラウザ側で javascript のライブラリを使って行うようにしてください。

※ アプリを構成する全てのファイルの影響箇所が更新され、必要なファイルが新たに生成されます

> ソートはブラウザ側で javascript のライブラリを使って行うようにしてください。

To update CsvDisplayController.java to handle sorting requests on the client-side using JavaScript, I'll modify the existing code as follows:

CsvDisplayController.java

```
@@ -1,5 +1,6 @@
// entire file content ...
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

+import com.opencsv.CSVReader;
+import com.opencsv.CSVReaderBuilder;
+
+import java.io.FileReader;
+import java.io.IOException;
+import java.util.ArrayList;
+import java.util.List;
+
@Controller
```

Next, I'll update csv.html to include a JavaScript library for sorting and add a form to submit the sort request:

csv.html

```
@@ -1,4 +1,5 @@
// entire file content ...
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Csv Display</title>
+  <script src="https://cdn.jsdelivr.net/npm/js-sequence@1.0.2/dist/sequence.min.js"></script>
+  <style>
+    table {
+      border-collapse: collapse;
+    }
+    th, td {
+      border: 1px solid #ddd;
+      padding: 10px;
+    }
+  </style>
</head>
<body>
  <h1>Csv Display</h1>
+  <table id="csv-table">
+    <!-- ... -->
+  </table>
+  <form action="/csv" method="post">
+    <select name="sortColumn">
+      <!-- ... -->
+    </select>
+    <input type="submit" value="Sort">
+  </form>
+  <script>
+    const table = document.getElementById('csv-table');
+    const sortButton = document.querySelector('input[type="submit"]');
+
+    sortButton.addEventListener('click', (e) => {
+      e.preventDefault();
+      const selectedColumn = document.querySelector('select[name="sortColumn"]').value;
```

Finally, I'll update pom.xml to include the necessary dependencies for JavaScript libraries:

```
<dependencies>
  <!-- ... -->
  <dependency>
    <groupId>org.jsdelivr.web</groupId>
    <artifactId>js-sequence</artifactId>
    <version>1.0.2</version>
  </dependency>
</dependencies>
```

## AI とペアプログラミング (Aider, Ollama)

2.8. (日本語でチャット) Spring batch を使った pdf 出力アプリを生成

【指示】 Spring batch を使い、 csv ファイルを読んで pdf に出力するアプリケーションを作れ

```
(aider) PS C:\Users\remoteuser\Aider-repo> aider --no-git --model ollama/llama3.1 --chat-language japanese --no-show-model-warnings
```

Aider v0.56.0  
Model: ollama/llama3.1 with whole edit for  
Git repo: none  
Repo-map: disabled  
Use /help <question> for help, run "aider

> Spring batch を使い、 csv ファイルを読んで p

Spring  
Batch を使い、 csv ファイルを読んで pdf に出力するアプリケーションを作るには、いくつかのファイルが必要になります。まず、 pom.xml に依存関係を追加します。

pom.xml

```
@@ -1,13 +1,14 @@
-<dependency>
-  <groupId>org.springframework.boot</groupId>
-  <artifactId>spring-boot-starter-web</artifactId>
-</dependency>
-<dependency>
-  <groupId>org.springframework.boot</groupId>
-  <artifactId>spring-boot-starter-thymeleaf</artifactId>
-</dependency>
-<dependency>
-  <groupId>com.opencsv</groupId>
-  <artifactId>opencsv</artifactId>
-  <version>5.3</version>
-</dependency>
+<dependencies>
+  <dependency>
+    <groupId>org.springframework.boot</groupId>
+    <artifactId>spring-boot-starter-batch</artifactId>
+  </dependency>
+  <dependency>
+    <groupId>com.opencsv</groupId>
+    <artifactId>opencsv</artifactId>
+  </dependency>
+  <dependency>
+    <groupId>com.itextpdf</groupId>
+    <artifactId>itext7</artifactId>
+  </dependency>
+</dependencies>
```

次に、 CsvToPdfJobConfiguration.java というファイルを作成します。このファイルでは、 Spring Batch のジョブを定義し、 csv ファイルを読み取り、 pdf に出力する処理を実行します。

CsvToPdfJobConfiguration.java

```
import org.springframework.batch.core.Job;
import org.springframework.batch.core.Step;
import org.springframework.batch.core.configuration.annotation.EnableBatchProcessing;
import org.springframework.batch.core.configuration.annotation.JobBuilderFactory;
import org.springframework.batch.core.configuration.annotation.StepBuilderFactory;
import org.springframework.batch.core.launch.support.SimpleJobLauncher;
import org.springframework.batch.core.repository.JobRepository;
import org.springframework.batch.core.step.tasklet.Tasklet;
import org.springframework.batch.item.ItemReader;
import org.springframework.batch.item.file.FlatFileItemReader;
import org.springframework.batch.item.file.builder.FlatFileItemWriterBuilder;
import org.springframework.beans.factory.annotation.Autowired;
```

Aider の起動オプションに --chat-language japanese を追加

【注】 ドキュメントに指定値の説明が書かれていなかったためこうしましたが、 Python 3.11 のロケール設定に倣い ja\_JP が正しいのかも...

## AI とペアプログラミング (Aider、Ollama)

(前頁続き)

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import com.opencsv.CSVReader;
import com.itextpdf.text.Document;
import com.itextpdf.text.pdf.PdfWriter;

@Configuration
@EnableBatchProcessing
public class CsvToPdfJobConfiguration {

    @Autowired
    private JobBuilderFactory jobBuilderFactory;

    @Autowired
    private StepBuilderFactory stepBuilderFactory;

    @Bean
    public Job csvToPdfJob() {
        return this.jobBuilderFactory.get("csvToPdfJob")
            .start(csvToPdfStep1())
            .build();
    }

    @Bean
    public Step csvToPdfStep1() {
        return this.stepBuilderFactory.get("csvToPdfStep1")
            .tasklet(new CsvToPdfTasklet())
            .build();
    }
}
```

## AI とペアプログラミング (Aider、Ollama)

(前頁続き)

CsvToPdfTasklet.java というファイルを作成します。このファイルでは、csv ファイルを読み取り、pdf に出力する処理を実行します。

CsvToPdfTasklet.java

```
import org.springframework.batch.item.ItemReader;
import org.springframework.batch.item.file.FlatFileItemReader;
import org.springframework.beans.factory.annotation.Autowired;

public class CsvToPdfTasklet implements Tasklet {

    @Autowired
    private ItemReader<String[]> reader;

    @Override
    public RepeatStatus execute(StepContribution contribution, ChunkContext chunkContext) throws Exception {
        CSVReader csvReader = new CSVReader(new FileReader("input.csv"));
        String[] line;
        while ((line = csvReader.readNext()) != null) {
            // csv ファイルの内容を pdf に出力する処理
            Document document = new Document();
            PdfWriter.getInstance(document, new FileOutputStream("output.pdf"));
            document.open();
            for (String field : line) {
                document.add(new Paragraph(field));
            }
            document.close();
        }
        return RepeatStatus.FINISHED;
    }
}
```

最後に、application.properties というファイルを作成します。このファイルでは、Spring Boot の設定を定義します。

application.properties

```
@@ -1 +1,2 @@
-csv.file.path=/path/to/csv/file.csv
+spring.batch.job.enabled=true
+spring.batch.initialize-schema=never
```

これで、csv ファイルを読んで pdf に出力するアプリケーションが完成しました。

Tokens: 588 sent, 866 received.

pom.xml

Allow edits to file that has not been added to the chat? (Y)es/(N)o [Yes]:

## AI とペアプログラミング (Aider, Ollama)

(前頁続き)

```
pom.xml
Allow edits to file that has not been added to the chat? (Y)es/(N)o [Yes]:

CsvToPdfJobConfiguration.java
Create new file? (Y)es/(N)o [Yes]:

CsvToPdfTasklet.java
Create new file? (Y)es/(N)o [Yes]:

application.properties
Allow edits to file that has not been added to the chat? (Y)es/(N)o [Yes]:
Applied edit to CsvToPdfTasklet.java
Applied edit to pom.xml
Applied edit to CsvToPdfJobConfiguration.java
Applied edit to application.properties

CsvToPdfJobConfiguration.java CsvToPdfTasklet.java application.properties pom.xml
> |
```

※ 生成されたクラスの動作確認はしていません

### 2.8.1. (Aider に質問) 作ったアプリの実行方法を聞いてみる

質問モード (/chat-mode ask) に切り替え

【質問】 このアプリケーションを実行するにはどうすればいい? …調子がいいと日本語で教えてくれる

```
CsvToPdfJobConfiguration.java CsvToPdfTasklet.java application.properties pom.xml
> /chat-mode ask

Aider v0.56.0
Model: ollama/llama3.1 with ask edit format
Git repo: none
Repo-map: disabled
Added CsvToPdfJobConfiguration.java to the chat.
Added CsvToPdfTasklet.java to the chat.
Added application.properties to the chat.
Added pom.xml to the chat.
Restored previous conversation history.

CsvToPdfJobConfiguration.java CsvToPdfTasklet.java application.properties pom.xml
ask> このアプリケーションを実行するにはどうすればいい?

Spring Batchジョブを実行するには、以下の手順を実行してください。

1 アプリケーションを起動します。たとえば、mvn spring-boot:run マンドを実行すると、アプリケーションが起動されます。
2 Spring Bootアプリケーションのログに「ジョブ実行中」というメッセージが出力されるはずです。これは、Spring Batchジョブが実行されていることを示しています。
3 ジョブの実行が完了すると、「ジョブ完了」というメッセージが出力されます。

このアプリケーションでは、input.csvファイルを読み取り、その内容をPDFに出力するジョブが定義されています。したがって、input.csvファイルが存在していることを確認してください。

また、アプリケーションのログに「PDF出力完了」というメッセージが出力されるはずです。これは、ジョブの実行が完了し、PDFが出力されたことを示しています。

Tokens: 1.7k sent, 312 received.

CsvToPdfJobConfiguration.java CsvToPdfTasklet.java application.properties pom.xml
ask> ログはどこに作られますか?

ログはSpring Bootアプリケーションの標準出力に出力されます。たとえば、mvn spring-boot:run マンドを実行すると、アプリケーションが起動し、ログが出力されるはずです。

ただし、ログの出力先を変更したい場合は、application.propertiesファイルに以下の設定を追加してください。

logging.path=/path/to/log/directory

この設定により、ログは指定されたディレクトリに出力されます。

Tokens: 2.0k sent, 154 received.

CsvToPdfJobConfiguration.java CsvToPdfTasklet.java application.properties pom.xml
ask> |
```

## AI とペアプログラミング (Aider、Ollama)

### 3. Aider のインストール／環境作成

Aider は Python (ver3.9 以上) で動作し、Aider の中から Git の操作 (commit、diff 他) ができることがウリになっています。LLM は各種のモデルが使えますが、ここではローカル環境で動作する Ollama を使う例を説明します。どのソフトウェアもライセンスは商用利用可能になっていますが、念のため各自各ソフトウェアのライセンス条項を確認してください。

上記のソフトウェアは winget (最新化済の Windows 10、Windows 11 であれば入っている) を使ってインストールできるので、このコマンドによる実行例を示します。

#### 3.1. Python 3

バージョン 3.9 以上が環境にない場合は、以下の手順でインストールします。

① 最新バージョンの確認

```
winget search --id Python.Python
```

② インストール先を指定してインストール

```
winget install -e --id Python.Python.3.12 --location C:#Tools#Python312
```

オプション

--id: 末尾の 3.xx を①で探した最新のバージョンにします

--location: パスに空白や日本語を含まないディレクトリを指定します

#### 3.2. Git

Git を使わないオプションで Aider を起動しても git.exe が呼出し可能であることを確認されるので、環境になればインストールが必要です。

① インストーラをダウンロードし会話形式で起動

```
winget install --id Git.Git -e --source winget --interactive
```

オプション

--interactive: 会話形式のインストールを行う

② インストーラへの応答

- ・ Select Destination Location で空白や日本語を含まないインストール先を指定します。
- ・ 他はデフォルトで構いませんが、Adjusting your PATH environment で

「Git from the command line and also from 3rd-party software」が選択されているのを確認

③ ターミナルを再起動して PATH を確認

再起動後のコマンドプロンプトから PATH、PowerShell からであれば \$ENV:Path を実行し、パスの中に Git のインストール先 %cmd が含まれていることを確認します。

#### 3.3. Ollama と Llama 3.1

Ollama は LLM を操作するインタフェースを提供してくれます。Ollama をインストール後に Ollama を使ってオープンソースの LLM Llama 3.1 をダウンロードします。

## AI とペアプログラミング (Aider、Ollama)

Llama3.1 は Aider のドキュメントで推奨<sup>2</sup>されていた Llama 3 の最新版で、ライセンスについては商用利用が許可されていますが、【LLAMA 3.1 コミュニティライセンス契約-2. 追加の商業条件】に以下の条件が記されているので、世界規模のビジネスを計画している場合は注意が必要です。

『前暦月の月間アクティブユーザー数が7億人を超える場合、お客様は Meta にライセンスをリクエストする必要があります。』

### ① Ollama のダウンロード

```
winget install -id Ollama.Ollama -e
```

※ 実行が完了すると ollama コマンドと ollama server がインストールされ、server はポート 11434 に常駐するようになります。Aider はこの 11434 ポートを使って Ollama と接続します。

### ② LLM (Llama 3.1) のダウンロード

Llama 3.1 はパラメータ数の異なる llama3.1:8b と llama3.1:70b があり、Aider はより高機能な 70b を推奨していますが、使用マシンに 40.8 GiB 以上のメモリがないと動作できません。

ここでは以下のコマンドで 8b をダウンロードします (#以降はコメントです)。

```
ollama pull llama3.1 #デフォルト 8b
```

### ③ ダウンロード済の LLM 一覧

コマンド ollama list で確認できます。

```
PS C:\Users\remoteuser> ollama list
NAME                ID                SIZE    MODIFIED
llama3.1:70b        c0df3564cfe8     39 GB   2 hours ago
llama3.1:latest     f66fc8dc39ea     4.7 GB  3 days ago
```

### ④ Ollama から LLM と会話する場合

```
ollama run モデル名
```

```
PS C:\Users\remoteuser> ollama run llama3.1
>>> あなたはJavaのアプリケーションを作ることができますか？
もちろんです。ここではSpring Bootを使ってJavaで簡単にWebアプリケーションを起動する方法について説明します。
### 環境
このチュートリアルでは、以下の環境が必要です。
* Java 11
* Spring Boot 2.3.0.RELEASE
### Mavenプロジェクトを作成
これで、Spring Bootアプリケーションが完成しました。
### まとめ
このチュートリアルでは、Javaのアプリケーションを作成し、Mavenプロジェクトを作成する方法について説明しました。また、依存関係を追加し、Hello Worldアプリケーションの実装も行いました。
----
このチュートリアルの場合、マイクロサービスアーキテクチャの例ではありません。
>>> Send a message (? for help)
```

※ Llama 3.1 のサポート言語に日本語は含まれませんが、まあまあ通じる日本語が返ります

※ 上は饒舌な回答の例ですが、同一質問に対する回答でもその都度異なり“Java は知りません”だったりエラーコードを含む回答になる場合があります、Aider がソース検証と再問合せを行います

<sup>2</sup> LLM への接続-Ollama <https://aider.chat/docs/llms/ollama.html>

## AI とペアプログラミング (Aider、Ollama)

### 3.4. Aider 動作環境の作成

Aider は Python のモジュールとしてインストールします。依存モジュールを多数使うので、他のモジュールとバージョンの衝突を起こさないように Aider 専用の Python 仮想環境を作ります。

仮想環境ではその環境のバージョンに合った Python のモジュールが使えるように環境変数 PATH が書き換えられます。以降の例は、次の環境条件を前提にしています。

- ・ホームディレクトリ配下に `pyvenv#aider` という環境を作る
- ・コマンドの実行は、ホームディレクトリ (`cd ~`) から行う

#### ① 仮想環境作成

```
py -m venv .#pyvenv#aider
```

#### ② 仮想環境活性化

```
powershell.exe -ExecutionPolicy ByPass -NoExit -Command "~#pyvenv#aider#Scripts#activate.ps1"
```

- ※ コマンドは 1 行で入力するか、改行したい場合は前行末に ` (バッククオート) をつけてください
- ※ 仮想環境で `deactivate` と打鍵すると、仮想環境から抜けます (PATH が書き戻される)

#### ③ Aider のインストール

```
pip install aider-chat #仮想環境ではコマンドの py -m は不要です
```

- ※ パッケージ名は“aider”ではなく、“aider-chat”です

#### ④ 環境変数 OLLAMA\_API\_BASE に Ollama server の URL:ポートを設定

```
setx OLLAMA_API_BASE http://127.0.0.1:11434
```

- ※ ollama server がインストールされると、ポート 11434 に常駐します
- ※ `setx` コマンドを使うと永続的にユーザ環境変数に設定され、システムの詳細設定[環境変数]で消すまで残ります

### 3.5. Aider の起動

Aider は仮想環境を使い、編集対象の資材が格納されているディレクトリで作業します。

#### ① Aider をインストールした Python 仮想環境 (前項で作成した仮想環境) を活性化

```
powershell.exe -ExecutionPolicy ByPass -NoExit -Command "~#pyvenv#aider#Scripts#activate.ps1"
```

#### ② ソースを保存しておく作業ディレクトリ (Git を使う場合はリポジトリ) に移動する

```
cd 作業ディレクトリ
```

#### ③ LLM を指定して Aider を起動する …チャットが始まります

```
aider --no-git --model LLM モデル名
```

[Aider の起動オプション]

- `--no-git`: Git を使わない。但し、Git のインストールは必須です (Aider v0.56.0 の場合、以下同)
- `--model`: LLM は各種使えますが、この例ではローカルに LLM を用意しています
- `--chat-language`: v0.56.0 で追加されたオプションで、`japanese` を指定すると不安定・部分的ながら日本語でチャットの応答を返してきます

## AI とペアプログラミング (Aider、Ollama)

```
(aider) PS C:\Users\remoteuser> aider --no-git --model ollama/llama3.1  
Aider v0.56.0  
Model: ollama/llama3.1 with whole edit format  
Git repo: none  
Repo-map: disabled  
Use /help <question> for help, run "aider --help" to see cmd line args  
> |
```

以上