

# 俺の強つよAI利活用な事例／ TIPSを共有しまくる会

## 講師紹介



**コーディネーター**

**山本健太**

**株式会社デンソークリエイト**



**コーディネーター**

**及川達裕**

**近畿職業能力開発大学校**

## 本セッションの進め方

皆さんAIは活用していますよね？

ちょっとした工夫や、がっつり使い倒すなど使用方法はひとそれぞれ  
本セッションでは、AIのおすすめの活用法を語りあいましょう！！

1. 進め方の説明(5分)
2. 活用法紹介(1人5分×4人…ですが、熱量に応じて延長予定)
3. ディスカッション(5分前まで)
4. まとめ(5分)

## 発表順

| 順番 | 氏名    | タイトル                                     |
|----|-------|------------------------------------------|
| 1  | 及川達裕  | 対話型AI使って便利だったTIPS！n連発                    |
| 2  | 岩崎雅弘  | もうNotionじゃねえ、無料枠尽きたんでな、今はObsidianだ       |
| 3  | 笹竹 佑太 | AIを「使う」から「組み込む」へ<br>ー PRレビューと自分専用サービスの構築 |
| 4  | 細合晋太郎 | 最近作ったOSSと作り方                             |

**それでは、さっそく紹介へ行きましょう！！**

# 対話型AI使って便利だったTIPS！ n連発

近畿職業能力開発大学校  
及川 達裕

# みんなAIリテラシ高くね？

- 私のポジペ

対話型AI をちょこちょこ使ったり，ローカルLLM にキャッキャしてたら，あっという間にAI エージェントが台頭してきて驚きです．変化が速すぎる…と思わないでもないですが，AI エージェントで楽しそうに遊んでる人がチラホラいるので，SWEST28 をキッカケにブーストをかけて仲間入りしたいところ．

- AIエージェントも良いけど，すぐに誰でも使える  
対話型AIも扱いたいよね（本セッション的な意味で
- 今回扱うもの
  - Gemini/ ChatGPT / NotebookLM

# ① コードをリファクタリングする前に UMLを吐かせてAIの認識との位相合わせ

どんな場面？

1. コードをPythonでガシガシと勢いで書いて動作確認
2. Geminiにコードと一緒にクラス化の指針を伝える
  - ポイント：  
「～をPlantUML形式で出力して」
3. UMLを確認後，OKでならコード生成
4. イマイチなUMLなら修正点を与えてもう一度😊



Mermaid 記法で吐き出す流派もあるらしい

## ② NotebookLMにデータシートを食わせて、 周辺ペリフェラルの扱い方のあたりをつける

- LLMはRAMをとにかく消費
- 容量の大きいファイルは全文を探索しない
- NotebookLMは、PDFファイルを全文検索してくれる
- データシートや製品マニュアルを読む際にあたりをつけるのに便利



参考：RX62Nマイコンのハードウェアマニュアル [1977ページ](#)



+ ソースを追加

ウェブで新しいソースを検索

🌐 ウェブ ▾

🔍 Fast Research ▾

🔍

☰

すべて選択 ☒

|                                                                                                               |                                     |
|---------------------------------------------------------------------------------------------------------------|-------------------------------------|
|  (2026-07-26改訂)実践組込み技術講座.pdf   | <input checked="" type="checkbox"/> |
|  11_RX62N_ハードウェアマニュアル.pdf      | <input checked="" type="checkbox"/> |
|  31_EMB-RXIO-62N_ハードウェア仕様書.pdf | <input checked="" type="checkbox"/> |
|  32_EMB-RXIO-62N_チュートリアル.pdf   | <input checked="" type="checkbox"/> |

このマイコンのAD変換器の初期化方法を教えて。どんなレジスタをどう叩いたら良い？

● Thoughts ▾

RX62Nマイコンに内蔵されている\*\*12ビットA/Dコンバータ（S12AD）\*\*を初期化し、例えば温度センサが接続されている **ANO端子（PORT4のP40）** からデータを取り込むための手順と、設定すべきレジスタの叩き方を解説します。

A/D変換を開始するには、大まかに\*\*「S12ADへの電源（クロック）供給」「ピンのアナログ入力設定」「A/D変換動作モードの設定」「トリガ（起動要因）の設定」\*\*というステップを踏みます

1 ... 。

**A/D変換初期化コードの例（温度センサ・ANO使用時）**

以下は、タイマ（MTU0）のコンペアマッチトリガを契機に、1秒間に1000回自動でAD変換をスタートさせ、変換終了時に割り込みを発生させる設定例です 1 ... 。

```
#include "iodefine.h"

void initAD(void)
{
    /* 1. S12ADモジュールのストップ状態を解除（クロック供給開始） */
    MSTP(S12AD) = 0;          // 12bit AD変換回路動作 [1]

    /* 2. 使用するピン（ANO / P40）をアナログ入力用に設定 */
    PORT4.DDR.BIT.B0 = 0;      // AN0を使用するためデジタル入力を無効化（入力モードに設定） [2]

    /* 3. A/Dコントロールレジスタ（ADCSR）の設定 */
    S12AD.ADCSR.BIT.CKS = 3;    // 変換クロック選択：PCLK/1 [2]
    S12AD.ADCSR.BIT.EXTRG = 0;  // 外部トリガではなく、ADSTRGRレジスタで選択したタイマ要因によるトリガ [2]
    S12AD.ADCSR.BIT.TRGE = 1;   // 外部トリガ、MTU、TMRトリガによるスキャン変換を許可 [2]
    S12AD.ADCSR.BIT.ADIE = 1;   // スキャン終了後の S12ADIO割り込み発生を許可 [2]

    /* 4. A/Dチャンネル選択レジスタ（ADANS）の設定 */
    S12AD.ADANS.BIT.ANS = 0;    // AN0（ポート4のP40）を選択したチャンネル [2]
```

### ③ 市販の電子部品を探してもらう

細合先生から教えても

- 電子部品の選定
- 各販社を列挙
- めんど
- Gemin 当に楽

CR2032専用の縦置き電池ボックスを探してる。

そうそう、DIP基板取付タイプが良いです。価格も一緒に

そこの表にリンクも含めて



実装)



タイプの価格と型番です。

| ショップ | 型番               | 特徴            | 税込価格 | 商品ページリンク                |
|------|------------------|---------------|------|-------------------------|
| 秋月電子 | CH74-2032LF      | 省スペース縦型（小型）   | ¥75  | <a href="#">秋月電子で見る</a> |
| 秋月電子 | CH243-2032LF     | 標準的な縦型 DIPタイプ | ¥90  | <a href="#">秋月電子で見る</a> |
| マルツ  | CH74-2032LF      | 省スペース縦型（小型）   | ¥82  | <a href="#">マルツで見る</a>  |
| マルツ  | CH243-2032LF     | 標準的な縦型 DIPタイプ | ¥112 | <a href="#">マルツで見る</a>  |
| マルツ  | GB-BHV-2032RU-MR | 縦型 DIPピン取付用   | ¥275 | <a href="#">マルツで見る</a>  |

税込価格

縦型

¥75

の

型

¥90

取付用

¥69

マルツ

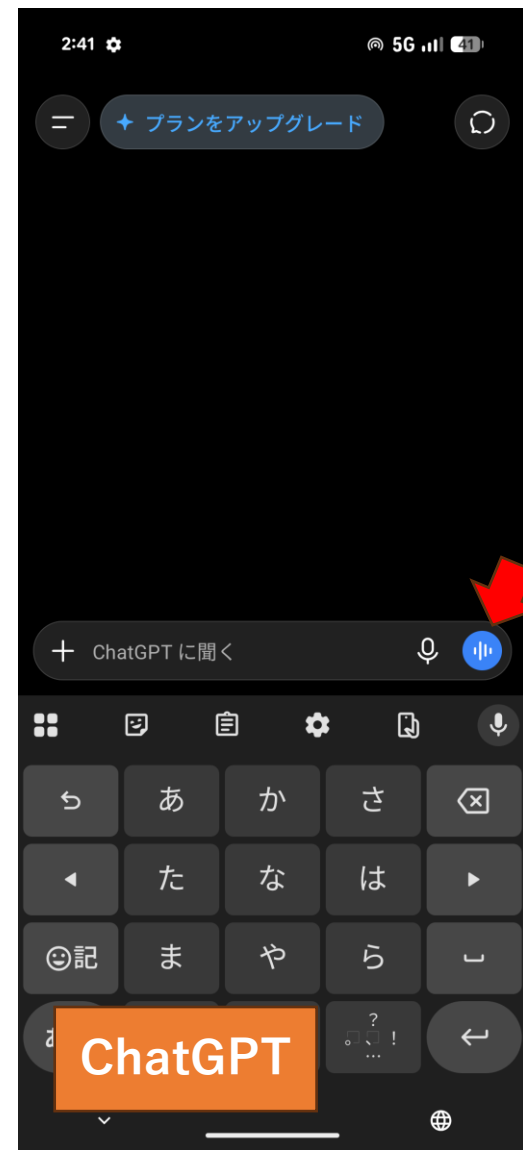
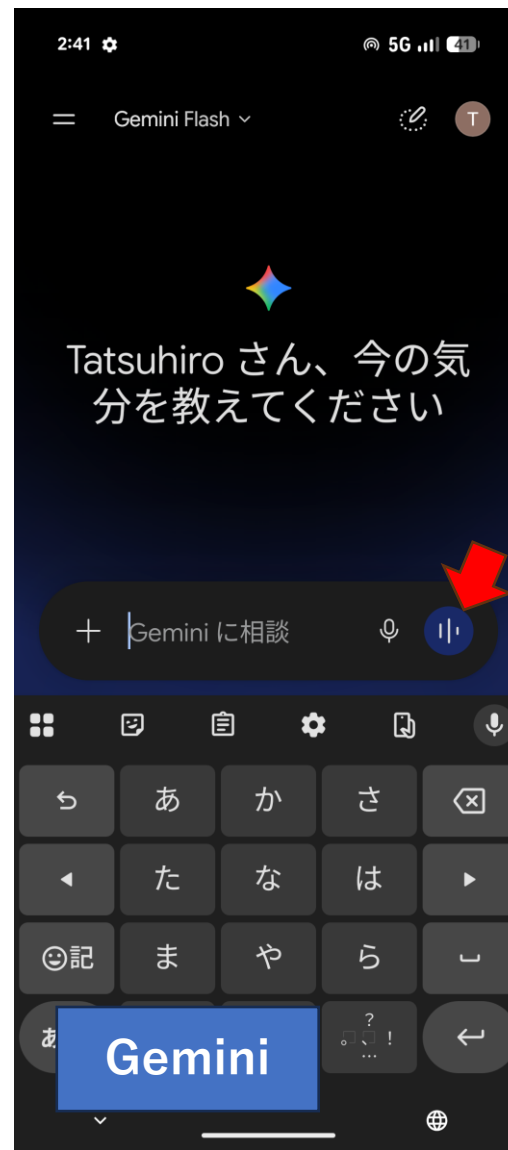
※マルツで販売されている CH74-2032LF や CH243-2032LF はタカチ電機工業扱いなどの  
同等品型番もあり、取扱ブランドによって数十円ほど価格が前後する場合があります

## ④ 通勤時に音声対話モードで壁打ち



通勤時の15～20分の有効活用

- スマホを車のマイク & スピーカーに繋いで
  - Gemini Live起動して、Geminiと会話しながら運転
  - ChatGPTだとAdvanced Voice Mode
- よくやること
  - 知らない技術領域の下調べ
  - 検討しているライブラリやフレームワークの調査 など
- 内部的には音声→テキストに変換して、テキストをLLMへ投げる
  - よって通信量は大したことは無さそう
  - 音声→テキスト変換の精度は、ChatGPTの方が良い印象



# ログを後から追える

“なるほどね。でも、モダナスTCPって結局どういう通信プロトコルなの？  
もうちょっと詳しく教えて。”

もちろん！MODBUS TCPは、シンプルなマスタ・スレーブ型の通信プロトコルで、イーサネット上で動くことを前提にしているんだ。データは「レジスタ」と呼ばれる単位でやり取りされて、マスター（PCなど）がスレーブ（PLCなど）にリクエストを送って、スレーブがそのレジスタにアクセスしてデータを返す仕組み。標準化されているから、異なる機器間でも比較的簡単に使えるのが特徴だね。

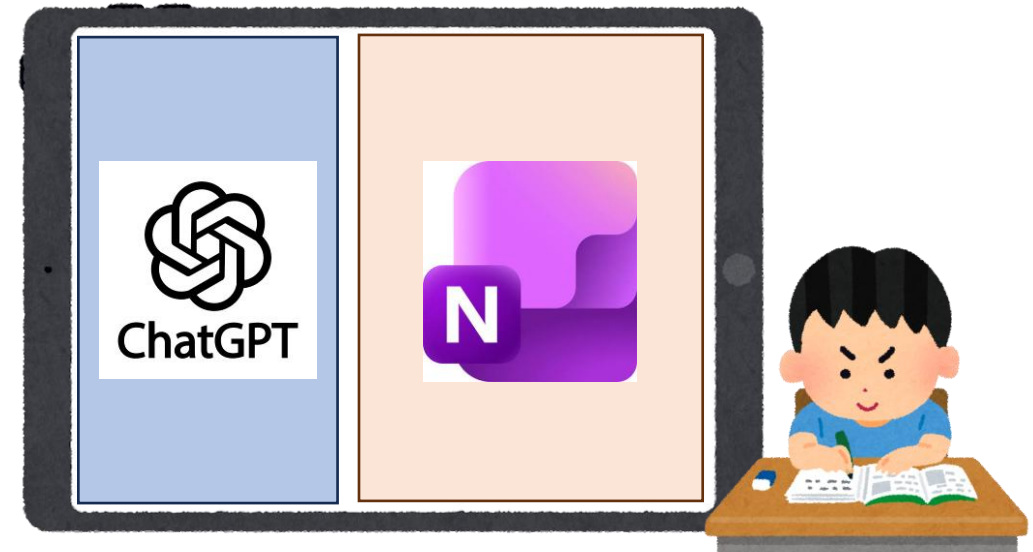
📄 📌 ... | 📄 PDFに変換

“なるほどね。ちなみに、ソケット通信でやると、さすがにレジスタを直接制御するみたいなことはできないのかな。”

そうだね、ソケット通信自体はあくまでデータのやり取りの枠組みだから、レジスタを直接制御する機能はないんだよね。だから、自分でプロトコルを設計して、どんなデータをどう送るかを決めないといけない。一方、MODBUSみたいなプロトコルを使えば、レジスタ単位でデータの読み書きができるから、そこが大きな違いだね。

📄 📌 ... | 📄 PDFに変換

- ChatGPTのログ事例
- テキストに起こされてログが残る
- 多少、変換ミスがあっても割と伝わる
  - 例：モダナスTCP → MODBUS TCP
- 関連事例：タブレットで壁打ちしながらメモをまとめる



以上！！

# ディスカッションタイムでは…

- こんな粒度のTIPSからでOK!! 共有しましょう
- その方法なら，こうするともっと良いよ！も募集
- 明日からサクッと使えそうなTIPSを共有しあえたら

もう Notion じゃねえ  
無料枠尽きたんでな、  
今は Obsidian だ

京都工芸繊維大学 岩崎雅弘

# 背景

## Before

- ChatGPTのNotion連携で壁打ち内容を保存
- 手順書やTODOを後から見返しやすい形にする
- ただし、AIは無料枠を気にせず長文を書き込む



便利だけど、  
無料枠が先に尽きる

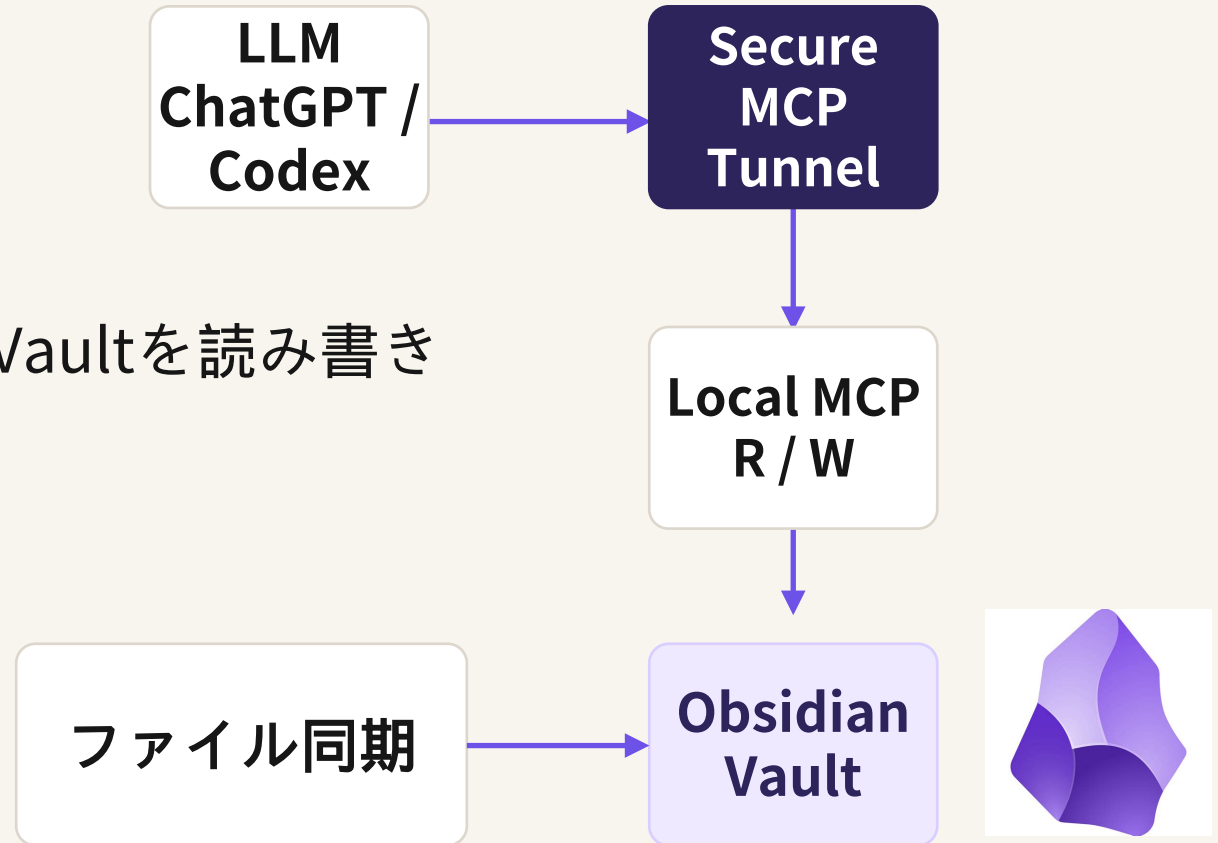
LLM  
長文生成

Notion  
無料枠

# Obsidianに寄せる構成

## After

- 保存先をMarkdownのVaultに変更
- SyncthingでPC・iPhone間を同期
- Tailscaleで端末間VPNを構成
- R/W可能なMCPを用意し、LLMからVaultを読み書き



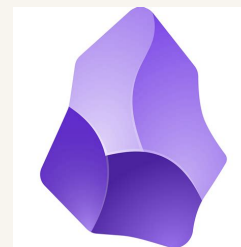
# 結果

- CodexからでもChatGPTからでも同じVaultに保存できる
- Notionの無料枠を気にしなくてよくなった
- 追加コストはほぼ電気代だけ

電気代以外  
タダ！

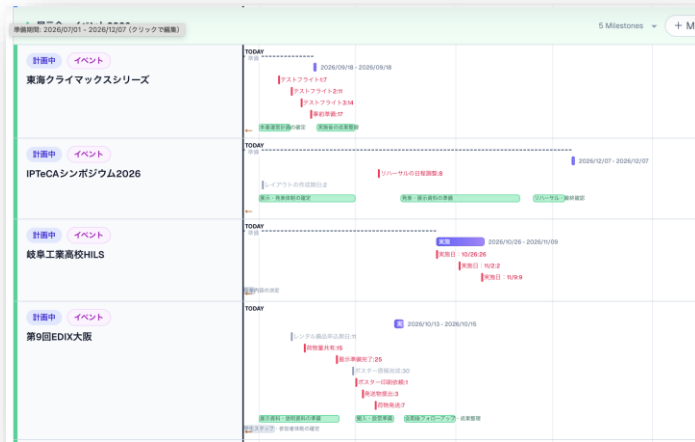
---

脱Notionを達成！



# AIを「使う」から「組み込む」へ — PRレビューと自分専用サービスの構築 —

1. AIを単発のツールとして「使う」だけでなく、日常のワークフローやサービスそのものに「組み込む」活用事例を紹介します。
2. AIに質問したり生成を依頼したりするだけでなく、AIを業務フローや開発基盤の一部として組み込むことで、仕事の進め方そのものを変えている事例を紹介します。



自分の使い方に合わせたWebサービス  
「Milestage」へ移行した事例



学生がGitHub Pull Requestで提出する  
Markdown報告書を、OpenAI APIを用いて自  
動チェックする仕組み



Dashboard

Overview

Timeline

Tasks

Calendar

ドキュメント

AIアシスタント

設定

忙しさ指数カレンダー

今後12ヶ月の日別負荷をヒートマップで表示。Googleカレンダーの予定とタスクの作業枠から実際の占有率を算出し、赤枠は期日までに終わらなそうなタスクがある日です。



2026-08-25 予定 6時間15分 / 8時間 締切タスク 1件 占有率 78%

● 空き ● 低 (<50%) ● 中 (50-75%) ● 高 (75-100%) ● 満 (100%+) ○ 間に合わなそうなタスクの締切 ● Googleカレンダーの予定 ● 作業ブロック

忙しさ指数

今後12ヶ月の月別負荷スコア。Milestone の重複数・KeyDate 件数を優先度/重要度で重み付けした値です。



今週の忙しさ指数

今週（月～日）の仕事時間に対する予定の占有率。作業ブロックと Google Calendar の予定（連携時）を合算しています。



OVERVIEW

Project / Milestone / KeyDate

Project を起点に、配下の Milestone と KeyDate を時系列順にまとめて確認・編集できます。

マインドマップ

ツリー

表示中データをDL

+ Project

+ Milestone

+ KeyDate

+ Task

終了済みを表示 (31)

Project

すべて

状態

状態でOR絞り込み

Milestone Tag

Milestoneタグで絞り込み

検索

Milestone・KeyDate を検索...

リセット

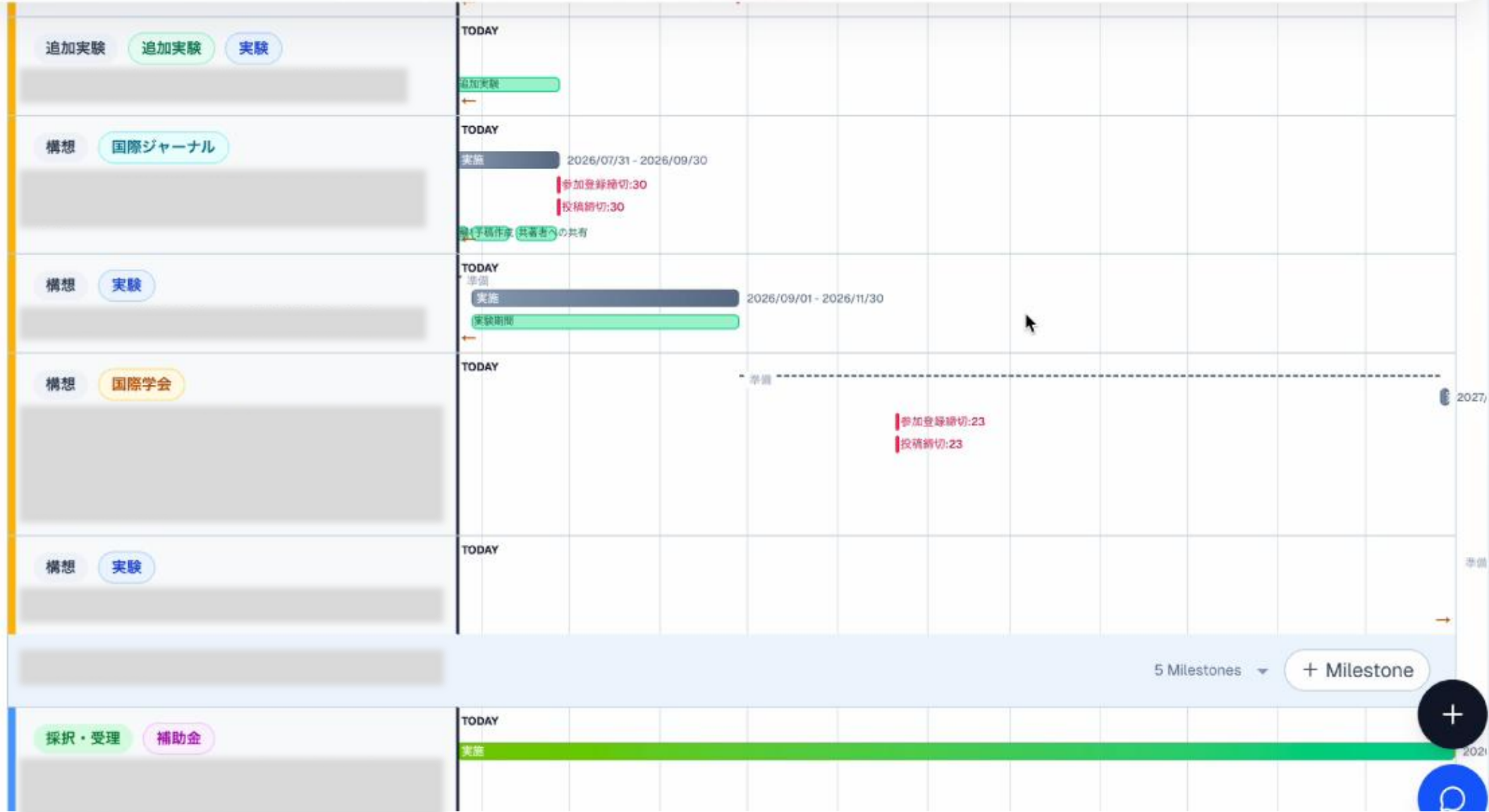
|   |  |            |            |
|---|--|------------|------------|
| > |  | 2026/02/08 | 2026/02/08 |
| > |  | 2026/02/08 | 2026/02/08 |
| > |  | 2026/02/08 | 2026/02/08 |
| > |  | 2026/02/08 | 2026/02/08 |
|   |  | 2026/02/08 | 2026/02/08 |
| > |  | 2026/02/08 | 2026/02/08 |
|   |  | 2026/05/13 |            |
| > |  | 2026/05/29 | 2027/03/31 |
|   |  | 2026/06/01 | 2027/03/31 |
| > |  |            |            |

+

ROADMAP TIMELINE  
Timeline

現在の設定をデフォルト保存

印刷



TASK BOARDS

Milestone / Task

Todo

通常 2h 2026/07/31 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 4h 2026/07/31 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 2h 2026/07/31 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 2h 2026/08/01 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 2h 2026/08/01 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

In Progress

通常 2h 2026/08/02 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 5h 2026/08/12 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 5h 2026/08/17 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 5h 実績 4h 2026/08/25 自動スケジュール

Todo

通常 5h 2026/08/26 自動スケジュール

△ 本来はすでに完了しているべきタスクです。

Todo

通常 5h 2026/08/27 自動スケジュール

△ 指定期間内の作業枠が足りません。

Todo

通常 5h 2026/08/28 自動スケジュール

△ 指定期間内の作業枠が足りません。

Todo

Todo

Todo



PLANNER

Task Calendar



更新

自動配置

Issues

61 tasks, 258h open

61

AIとリスクを相談する

未配置

未配置 5h 締切 2026/04/30

本来はすでに完了しているべきタスクです。

未配置

未配置 5h 締切 2026/06/15

本来はすでに完了しているべきタスクです。

未配置

計画対象外

13 tasks, 79.5h pending

13

未配置

未配置 1.5h 締切 2027/10/01

365日後より先の予定のため、未配置です。



今日



8/24(月) - 8/30(日)

09:00-18:00

25(火)

100%

8/26(水)

88%

8/27(木)

100%

8/28(金)

100%

8/29(土)

-

8/30(日)

-

SWEST28

SWEST28



ドキュメント

Wiki

DOCUMENTS

# ドキュメント台帳

Project / Milestone / KeyDate / Task に紐づく実施報告・計画のMarkdown文書一覧です。作成・編集は各エンティティの編集画面から行えます。

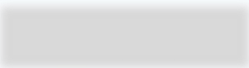
すべて

Project

Milestone

KeyDate

Task



Milestone: 小川財団・国際交流助成 2026/08/15 21:15





+ 新しい会話

⚠ リスケ相談を始める

プロジェクト一覧の提示

1日前・「SI2026 OS30 キーノート講...

⚠ 締切過ぎタスクの確認

1日前・現在のタスク状況に基づく...

今週の忙しさ予定

1日前・今週（2026年8月25日〜31日...

## AIアシスタント

予定の確認・追加・変更、忙しさの把握までまとめて相談できます

できること

### 📅 週間スケジュール

指定週の日別スケジュール（KeyDate・タスク・Googleカレンダー予定）と、日ごとの忙しさ（稼働率%）を表示します。

今週は忙しい？

来週の予定を見せて

明日って時間ある？

### 📊 月別忙しさチャート

今後12か月の月別の忙しさ（Milestone/KeyDateの重み + Googleカレンダー予定件数）をチャートで表示します。

今後の忙しさを教えて

繁忙期はいつ？

来期の負荷を見たい

### 📅 マイルストーン タイムライン

マイルストーンと配下のKeyDateを横棒のタイムラインで表示します（最大10件）。

Aプロジェクトの流れを見せて

今年度のマイルストーン一覧

この案件のスケジュール感は？

### 🕒 期限切れタスク

締切を過ぎたまま残っているタスクを、超過日数・残り工数・親のKeyDate（日付/確度度）付きで一覧表示します。

期限切れのタスクは？

リスケしたい

遅れているものを整理して

### 📝 リスケ案

タスクごとの新しい締切・作業開始日の提案をカードで表示し、ユーザーがその場で「この案で更新」を押して適用できます。

メッセージを入力...（Shift+Enterで改行）



+ 新しい会話

⚠ リスケ相談を始める

### 今週の忙しさの概要

たった今・今週は非常に忙しいです。...

#### プロジェクト一覧の提示

1日前・「SI2026 OS30 キーノート講...

#### ⚠ 締切過ぎタスクの確認

1日前・現在のタスク状況に基づく...

#### 今週の忙しさ予定

1日前・今週（2026年8月25日〜31日...

## 今週の忙しさの概要

予定の確認・追加・変更、忙しさの把握までまとめて相談できます

お手伝いできることがあれば教えてください。

### 📅 週間スケジュール

2026-08-25 ~ 2026-08-31



● Googleカレンダー ● High ● Medium ● Low ✓ タスク

#### 進行中のマイルストーン



メッセージを入力... (Shift+Enterで改行)



報告書の提出



Pull Request



AI  
Review



笹竹のチェック



提出完了



HiShi729 commented on Jul 17

...

### 変更内容

報告書システムの運用改善に関する 2026-07-16 の報告書と作業フロー図を追加します。

### 目的

README の提出ルールに従い、作業内容と検証・統合結果を記録します。



github-actions Bot commented on Jul 17

...

判定: NEEDS\_REVISION

#### 総評:

全体として目的と実施内容、統合手順が明確に書かれており、実装ブランチ・最終コミットID・作業フロー図まで記載されていて追やすい報告です。しかし、検証ロジックや運用ルールの具体仕様（正規表現、サンプル、エラーメッセージ例、既存



github-actions Bot commented 3 weeks ago

...

判定: SUGGESTION

#### 総評:

報告書は目的、実装、ビルド、ホストテスト、実機試験、結果まで一貫して記載されており、読者が実施した作業と得られた到達点を追える良好な構成になっています。テーブルや観測値、ログ配置の明示があり再現性確認のための情報も提供されています。ただし、一部の時間関連数値に読み手が混乱する恐れがあるため、明確化するとさらに読みやすくなります。

#### 修正必須:

なし



sassa4771 reviewed yesterday

[View reviewed changes](#)

sassa4771 left a comment

...

報告書が非常によくまとまっております。  
いくつかわからない点をコメントしたので、回答をお願いします。  
引き続きよろしくお願いします。

# 最近AIで作ったあれこれと その作り方

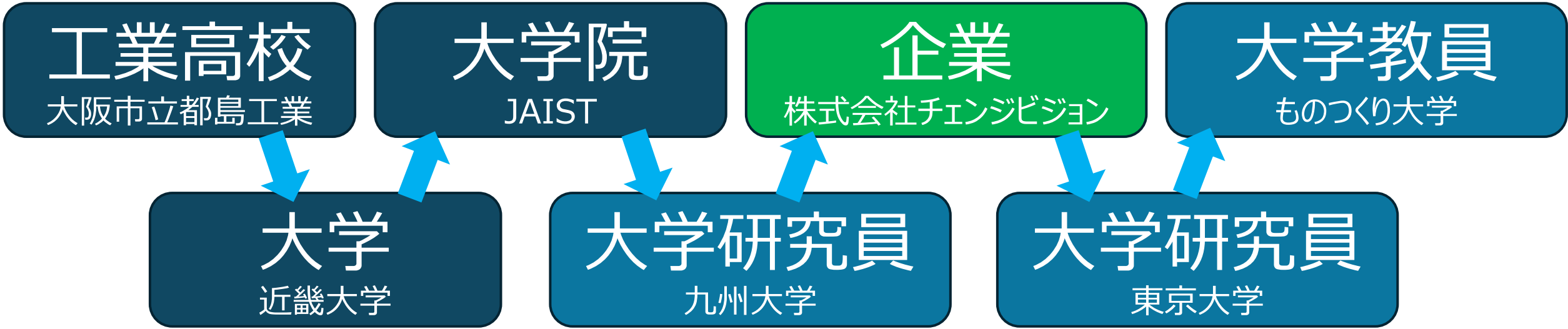
細合 晋太郎

ものづくり大学 技能工芸学部 情報メカトロニクス学科 准教授

ほそあい しんたろう

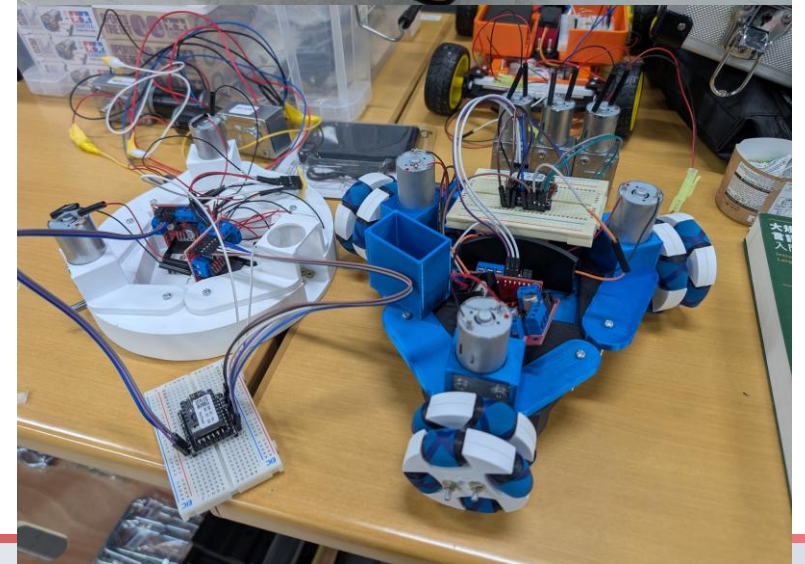
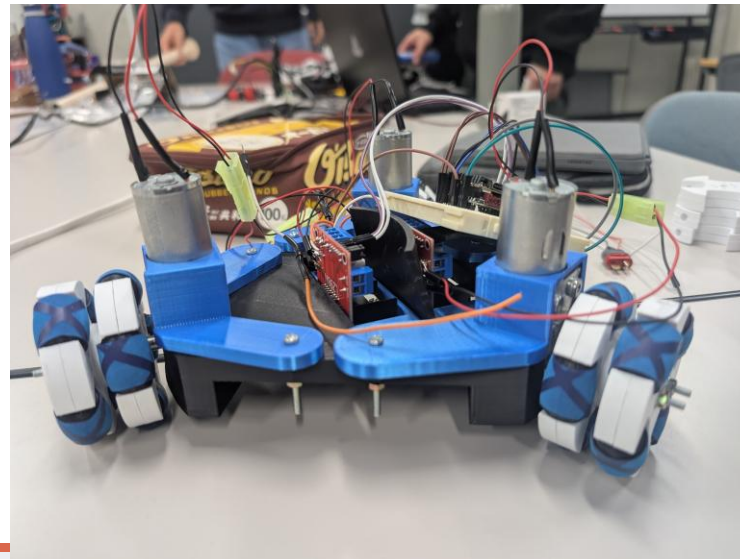
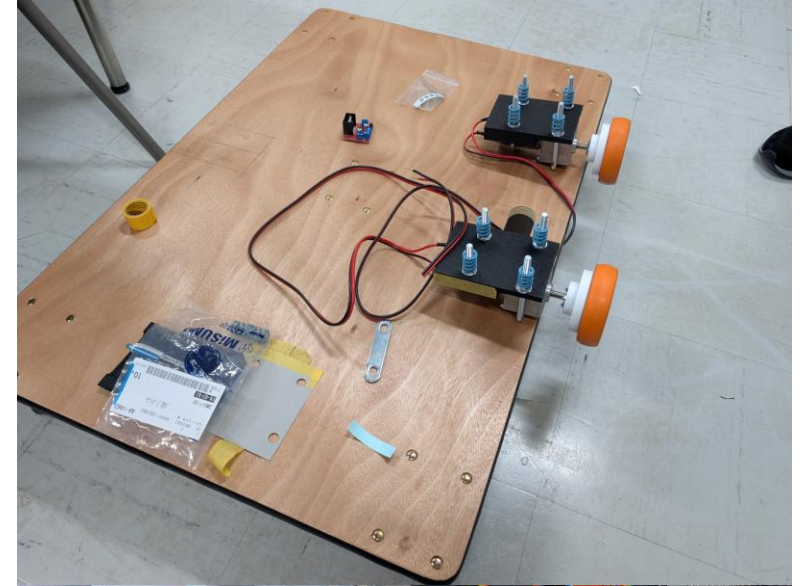
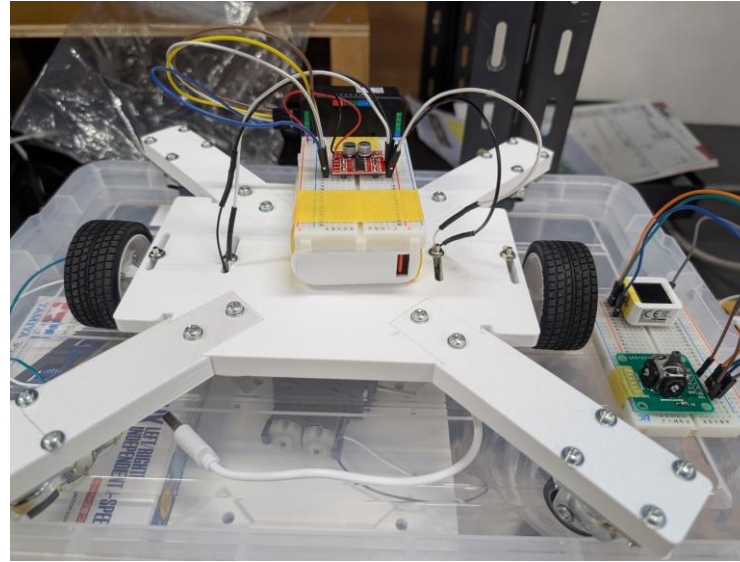
細合 晋太郎

略歴



専門分野：組込みシステム, IoT, ロボット

# いろんなロボットを作っています



# 利用している生成AI

Claude Pro

主力

Claude Code

Ollama

Gemma4 e2b, Ornith-1.5

LocalLLMおもちゃ

サブ

ChatGPT Plus

Codex

OpenAI API (微課金)

APIで遊ぶ用

Github Copilot

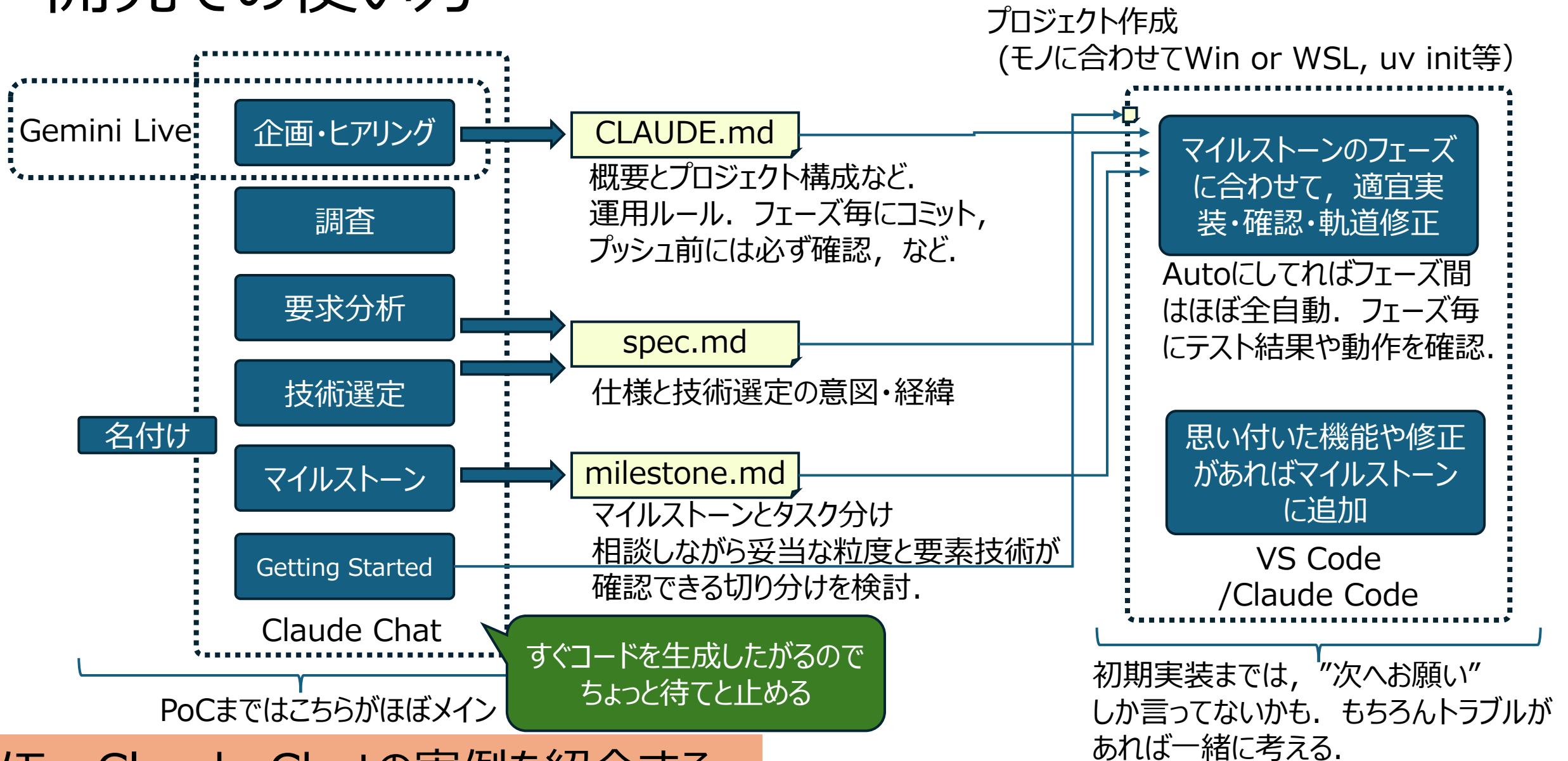
Gihubに課金したらなんかついてきてた。

Gemini (大学アカウント,  
個人アカウント (微課金))

Gemini Live

その他, 目についたら試すけど,  
定着は少ない..

# 開発での使い方～



メモ : Claude Chatの実例を紹介する

# 開発での使い方～

milestone.md

このマイルストーンで進められるのか，DoDで本当に確認できるのか，よく考える。（考えるのも修正もClaudeと一緒に）

## M1. Lチカまで

✓ 環境構築

✓ 素プロジェクトビルド

✓ Lチカビルド・焼き

実機確認 ✓ 人間

仕様が固まっていれば，  
環境構築も全自動．

TODOのDoDを確認する  
ためのテストも生成してもら  
う．時々CI

spec.md

うまくいかないところ，仕様を変更し  
たら，都度更新．設計意図も残  
す

## M2. ネットワーク疎通

□ PC-マイコン間 Ping

□ ライブラリで通信

□ MCU間通信

ひとつひとつ着実に，確実に動く  
コンポーネントを積み重ねる

idf.py flash / idf.py monitor等も自動で実行して，ログを読んで修正までしてくれるので，繋いでおけばok．

# FoamPath : 2D スタイロカッターCNCを作ってみたい

- ・ケース作るの大変・・, スタイロが任意形状に切れるといいかも! スタイロカッターCNCつくろ! ソフトは画像読み込み→2値・ベクタライズ→経路作成→G-Codeあたりがいろいろかな? ↓

ほぼノーミスで生成されたアプリのプロト.  
この後ハードとFirmwareも相談しながら作る予定

画像からベクターデータへ変換, その後一筆書きのGCodeへ変換する仕組みを検討しています. 2次元CNCでスタイロフォームのカットを行います.

スタイロフォームの2次元CNCカット用に、画像う流れですね。良いアプローチだと思います。具させていただきます。

今回、私に何をしてほしいですか？

パイプライン設計の相談

入力画像はどのようなものを想定していますか？

線画・ロゴ・文字、イラスト・図形

CNC側の制御・出力形式について、決まっていることはありますか？  
まだ未定

この用途だと、パイプラインは大きく3段階に分けて技術的な選択肢と注意点が変わってきます。

## 確定した仕様サマリー

入力はPNG/JPEG (SVGは将来対応)。画像は200mm×200mmのワークサイズ上に配置してスケール調整し、DPI情報があれば実寸換算の初期値として活用する。二値化は大洋の自動しきい値をデフォルトとしつつ、GUI上で手動調整可能にする。イラスト系入力は全体シルエットのみを残す方針で、これは今後も複雑なイラスト・写真は扱わない前提。

ベクター化はpotraceで輪郭抽出し、単一の最外周輪郭のみを採用（穴は無視）。パス簡略化の許容誤差、最小曲率半径・最小ブリッジ幅の閾値（違反時は警告表示のみ）は、いずれも初期値をプログラム側で妥当な値に設定した上で、GUI設定メニューから調整可能にする。

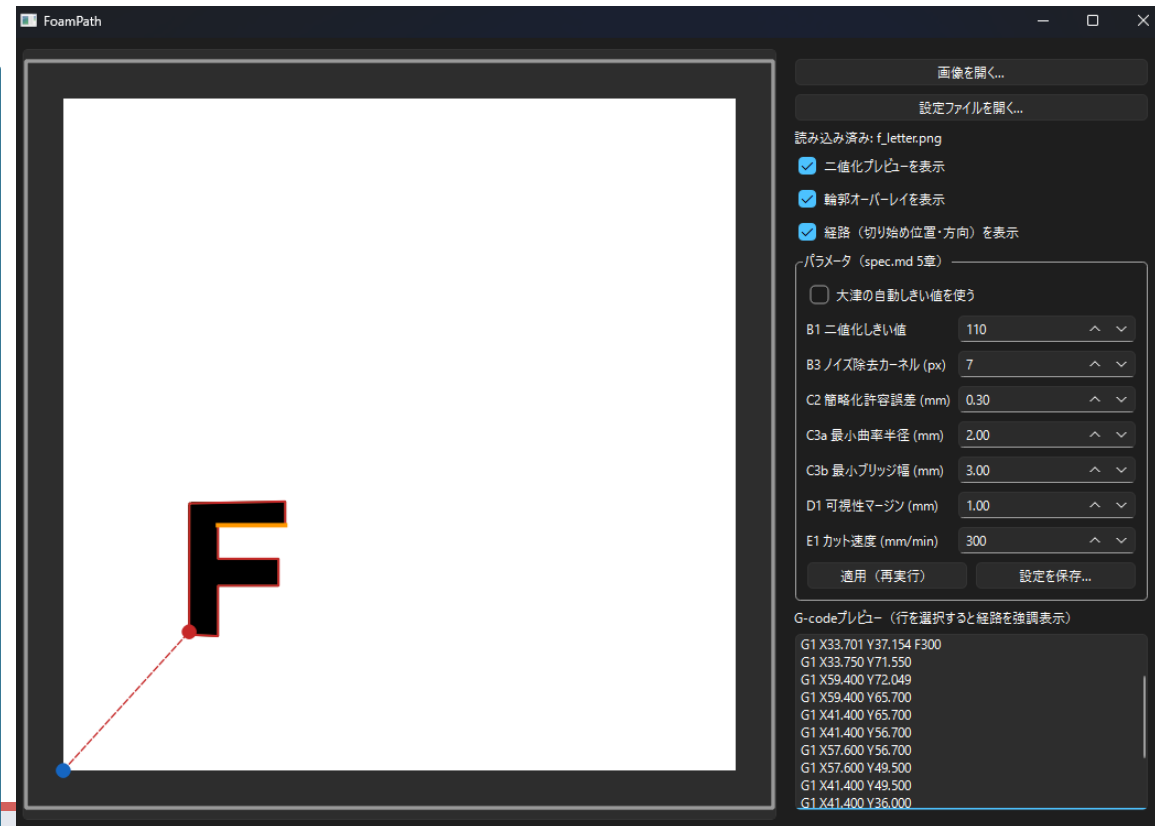
経路生成のMVPスコープは単一部品のみで、複数部品の順序最適化・可視性グラフによる部品間接続・手動再結線フローはすべて積み残し（ただし可視性グラフの膨張マージン値自体はGUI設定項目として先に用意しておく）。ワーク上の原点(10mm,10mm)にあらかじめ穴を開けてワイヤーを通していき、そこを基準として各部品の輪郭上の最近傍点を切り始め位置とする。

速度はカット区間・端材通過区間問わず一定値で、GUI設定メニューでCNCで一般的な単位 (mm/min) を使って指定する。ノイズ除去のモルフォロジーパラメータもGUI設定に含める。

GUIの機能範囲は「画像読み込み→サイズ調整→二値化プレビュー→輪郭抽出結果のオーバーレイ表示→経路の順序・方向表示→G-codeプレビュー」で、パラメータ変更後はボタンで再実行する形。技術選定はデスクトップアプリ想定で保留 (Web移行の可能性も視野)。

責務範囲はG-code出力まで (実機への送信は対象外)。作業エリアは200mm×200mm、原点 (10mm,10mm)。開発言語はPython (速度面で問題があればRust/eguiを検討)、コア処理ロジックはCLIとGUI両方から呼べるように分離する。

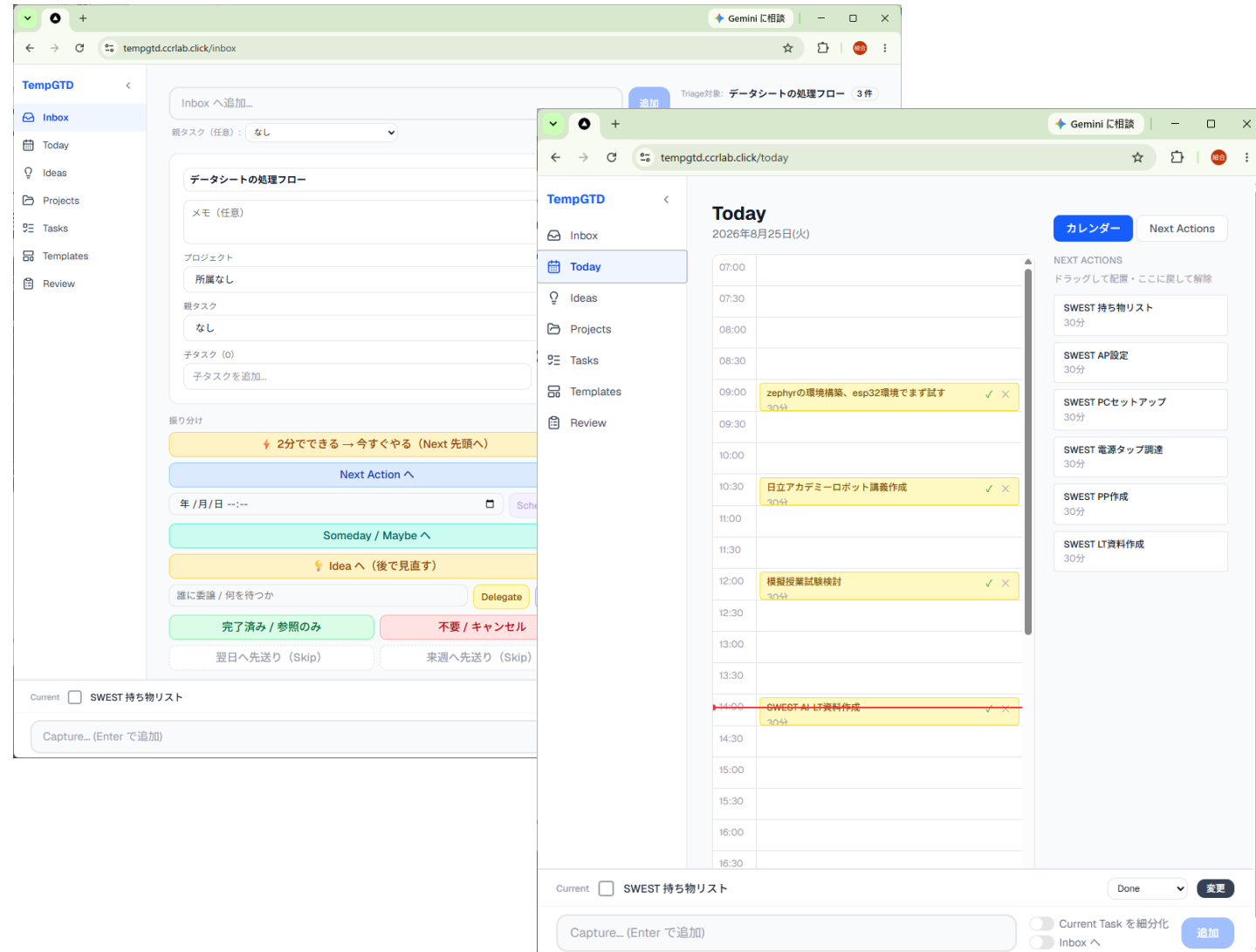
ここまでで一通りの仕様が固まったと思います。この内容で docs/spec.md・docs/milestone.md・CLAUDE.md の作成に進めてよいでしょうか、それとももう少し詰めた部分がありますか？



# TempGTD : 自分用タスク管理ツール

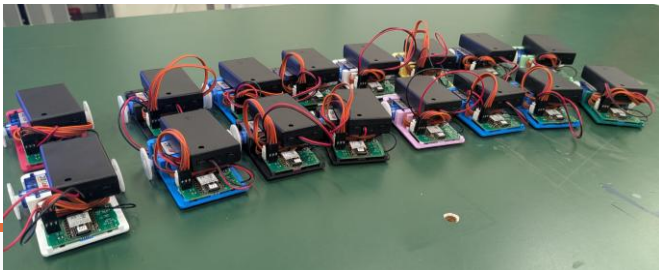
<https://github.com/s-hosoai/tempgtd>

- GTD手法に則ったツール
- 思い付いたことをじゃんじゃか Inboxへ詰め込む
- 定期的にトリージ（ざっくり捌く）． 週毎・月毎に棚卸
- 今日のタスクスケジュールに割り当て．あとは心を無にして取り組む
- 実はSWEST等のイベント運営用に作り始めたもの．  
Template GTDの意で，定型・類似のタスクセットを用意して，トリガーに応じて再実施を容易にしたい，という思いから始まったけど，普通使いになっている



# robot-blockly-kit : ESP32の実習を楽しみたい！

- オープンキャンパスや出張授業で、組み込みとかロボットの授業やってよと言われるが、「開発環境は当然入れられない，PCすらない」という状況でどうやって開発をさせろと・・・という呪詛から生まれたシロモノ.
- 「ESP32S3の上のサーバでBlockly動かして，Luaにトランスパイルすれば，スマホだけあれば組み込み出来ちゃうし，小中学生くらいでも触れるじゃん！？」

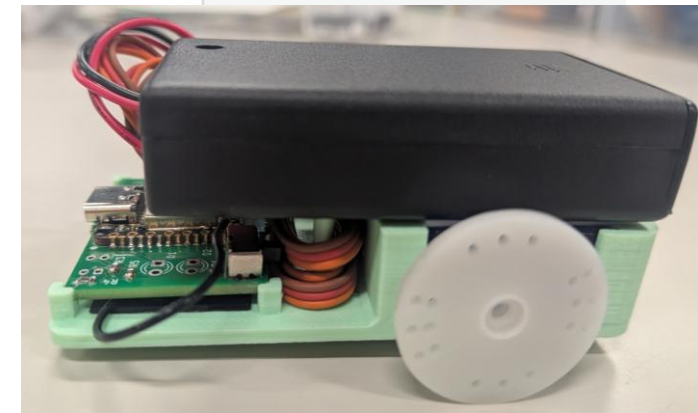


<https://github.com/s-hosoai/esp-blockly>

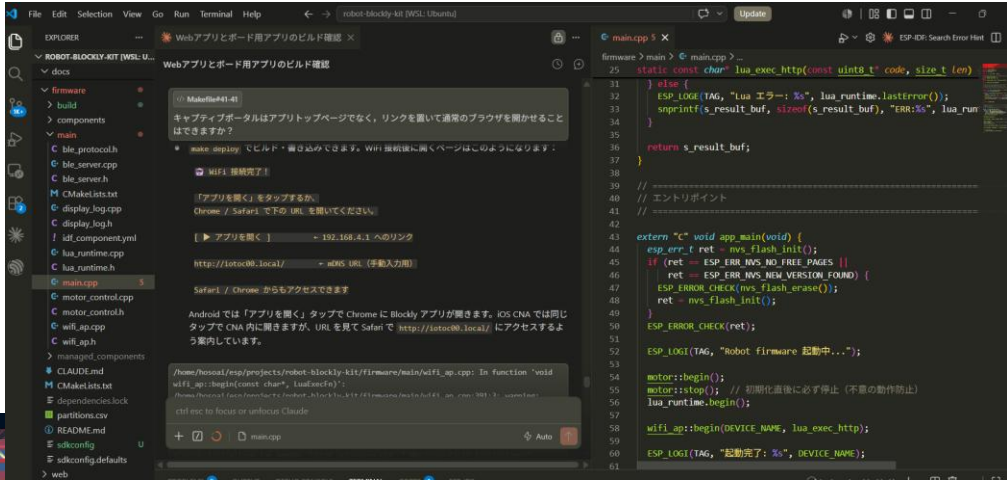
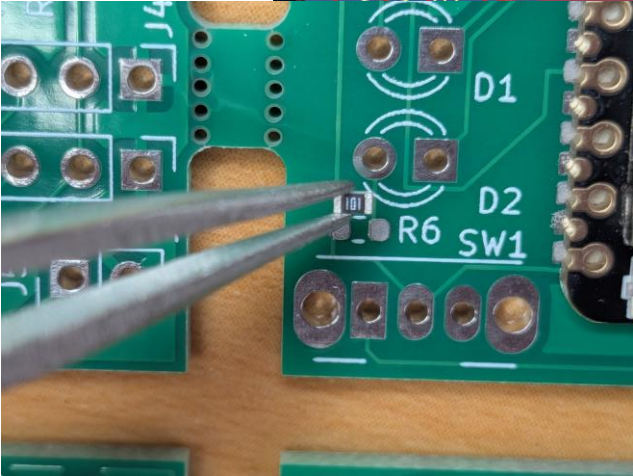
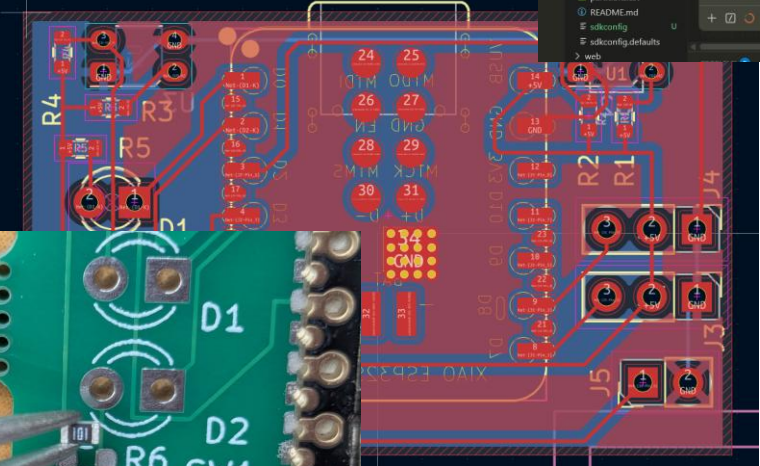
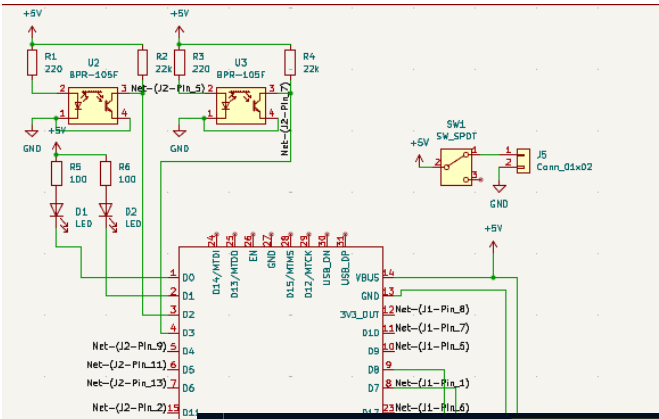
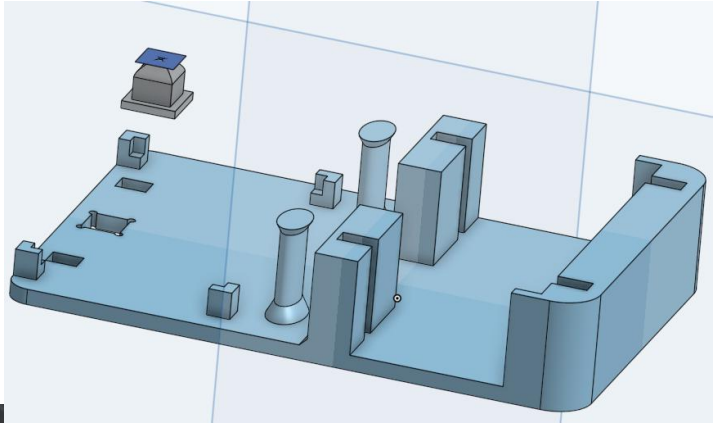
<https://github.com/s-hosoai/robot-blockly-kit>



| Blockly   |                |
|-----------|----------------|
| WebServer | Lua VM /driver |
| ESP32S3   |                |



# ハードもがんばった



### robot-blockly-kit

子供・初学者向けのロボットプログラミング学習キット。

Webブラウザ上でブロック (Blockly) を組み合わせることでロボットの動きをプログラムし、Bluetooth (BLE) 経由でロボット本体 (ESP32-S3) に転送して動かす。サーバーは使わず、ブラウザとロボットが直接通信する。

詳しい設計・前提・決事項は [docs/architecture.md](#) を参照してください。Claude Code で開発する際の指示は [CLAUDE.md](#) にまとめています。

### 構成

```
robot-blockly-kit/
├── web/      ブラウザアプリ (Vite + Blockly, バックエンド無し)
├── firmware/ ESP32-S3用ファームウェア (PlatformIO + Arduino + IaaS)
└── docs/    設計ドキュメント
```

### クイックスタート

#### Webアプリ (ブラウザ側)

```
cd web
npm install
npm run dev      # 開発サーバーを起動 (http://localhost:5173 など)
npm run build    # dist/ に静的ファイル生成 (GitHub Pages公開用)
```

詳細は [web/README.md](#)。

#### ファームウェア (ロボット側)

[PlatformIO](#) が必要です (VS Code拡張、または `pip install platformio` でCLI導入のどちらでも可)。

```
cd firmware
pio run          # ビルド
pio run -t upload # ESP32-S3に書き込み (実機接続時)
```

# その他

<https://github.com/s-hosoai?tab=repositories>

(散らかってます. すみません...)

- PDFStamper  
Adobeのサブスクしたくなくて秒で作った
- AngleMeter  
企業案件でESP32 + サーボx2の仰角計つくった.
- komaloom  
大学の時間割作成支援ツール (作りかけ)
- Aestra  
ユーザコンテキスト把握のため, スマホ・PC等の動態検知して通知するシステム (作りかけ)
- AtomVM Zenoh  
ESP32でAtomVM(Elixirの軽量実装)の上でZenoh-picoいごかした.
- ledtool  
NeoPixel系の発光パターンを作れるWebツール
- zenoh-esp-now  
ESP-NOW上でZenohいごかせるんじゃない? なんか知らんがいごいた.
- dux  
ディスク使用量確認ツール. duを使いやすくした感じ? Go スクリプトでサクッと導入できていい感じ

ソフトに限らずメカもエレキも  
思い付いたものをすぐに形できる  
とてもいい時代です  
どんどん作りましょう  
じゃんじゃん失敗しましょう  
何度もやんなきゃうまくならないのは、  
時代やツールが変わっても同じです。  
AIといっぱいお話をして、自分が楽しいものを  
いっぱい作っていきましょう！