

# CS集会

隔週火曜日で開催中!

ComputerScience集会 @VRChat

<<https://vrc.group/CSG.5184>>

# AI(LLM) powered なIDEを触ってみて - Kiro, Antigravity, Cursor, vs code(?) -

ComputerScience集会 @VRChat 2026/5/5

夜鍋ヨナ-yonabeyona

# 自己紹介

- 名前：夜鍋 ヨナ(よなべ よな)
- X(Twitter) : yonabeyona
- Discord : yona\_47
- その他
  - ComputerScienceが好き
  - 数学勉強中
  - 物理も勉強中
  - ComputerScienceの中でも、ComputerArchitectureが好き



# 今日する話

- ① AI(LLM) powered な IDE
- ② Antigravity の話 / ③ Kiro の話 / ④ Cursorの話
- ⑤ LLMトレンド論
- ⑥ VS Code の話
- ⑦ まとめ

# ① AI(LLM) powered な IDE

- Antigravity
- Kiro
- Cursor

## これらの特徴

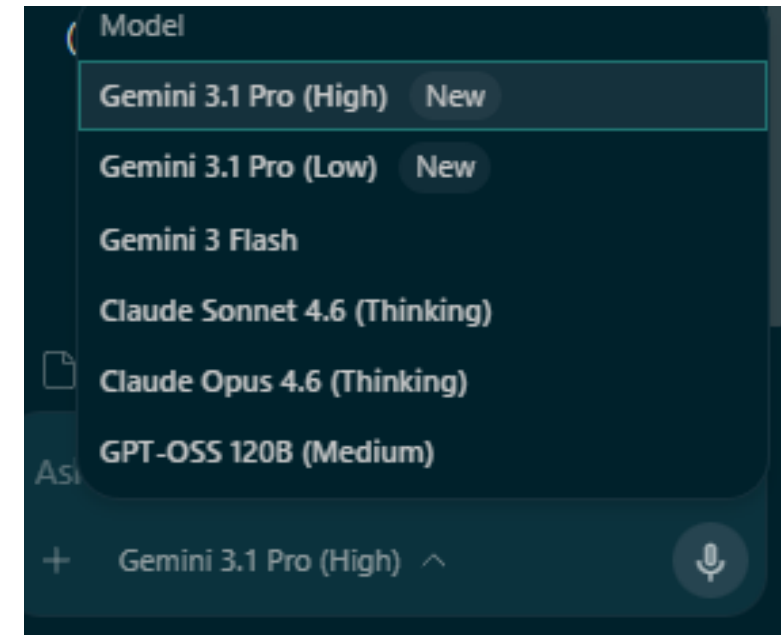
- VS Codeのフォーク製品
- AI(LLM)との連携が最初からされてる
- 各社が提供しているLLMプラットフォームと連携

## ② Antigravity の話

- Antigravityの特徴
- Antigravity感想
- Antigravityのウワサ

# Antigravityの特徴

- Google製で、Agent機能がウリ
- 主にGemini/Claude/GPT-OSSが使える
- BYOK非対応なので、OpenAIのAPI使えない
- 開発段階なので無料プランしか無い
- 個人プランは、入力データが学習に使われる
  - ちゃんと設定でオプトアウトしようね
  - Workspaceで利用したら、オプトアウトが標準になる模様



# Antigravity感想

- エージェント管理画面があり、複数エージェントに指示を出しやすい

- + New Conversation
- Conversation History
- Workspaces
- CS\_77
  - Understanding Gemini D... 9h
  - Understanding Antigravi... 9h
  - Analyzing Lightning Talk ... 9h
  - Analyzing Lightning Tal... 16h
  - Enhancing Lightning Tal... 16h
  - Updating Lightning Talk... 20h
  - Critiquing Lightning Tal... 20h
  - Committing Lightning T... 20h
  - Reviewing Lightning Tal... 21h
  - Fact-Checking LT Prese... 21h
  - Reviewing LT Presentati... 21h
  - Identifying Cursor Edito... 23h
- See less

このLTを見て、聴衆が受け取る印象と情報はどんなものがある?

Audience Impression Analysis

話題と、その枝葉についてまとめてみて  
どんなトポロジになりそうか確認したい

Worked for 2m

topic\_topology.md にまとめました。

### トポロジの要点

全体は「ダイヤモンド型」— 序盤で分岐 (Kiro/Cursor各論) → 中盤で合流 (トレンド論) → VS Code が合流して最終まとめ。

### 構造上の気になる点

箇所	懸念
④ Cursor の枝が3層まで深い	Cursor → Composer2 → Kimi K2.5 → 性能比較...と潜っていく、聴衆が「今何の話？」になるリスク
Antigravity の枝がほぼない	Kiro・Cursorに比べて薄く、4製品比較なのにバランスが偏る
④→⑤→⑥ で主語が3回変わる	Cursor → LLMトレンド → VS Codeと抽象度・主語が連続で切り替わる
② Kiro → ③ 仕様駆動の横断リンクが暗黙	CC-SDDがKiroの文脈とCursorの文脈の両方に関わるが、明示的に繋いでいない

特に ④ Cursorの深い枝 (Composer2→Kimi→性能比較→真逆の謎) を整理するかどうか、聴衆の迷子防止の鍵になりそうです。

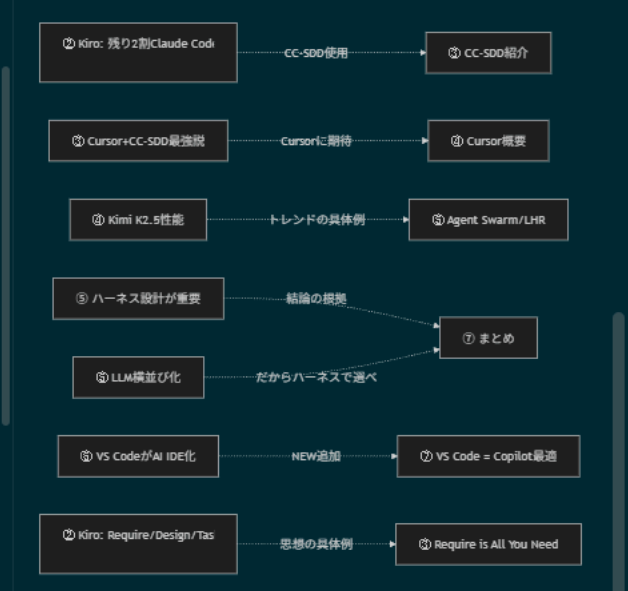
Topic Topology

Ask anything, @ to mention, / for workflows

+ Worktree Claude Opus 4.6 (Thinking)

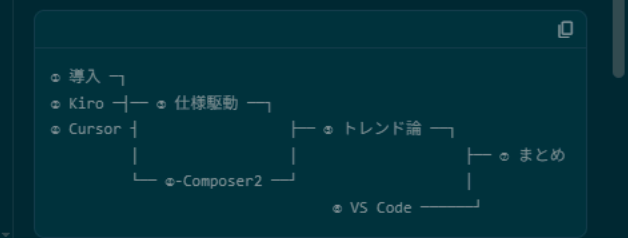
### 構造上の注目ポイント

「枝」が他の「幹」に合流する箇所 (横断リンク)



上図の点線が「暗黙の横断リンク」。聴衆はこの繋がりの中で補完する必要がある。

### 全体トポロジの形状



### 形状:「ダイヤモンド型」

- 序盤で分岐 (Kiro / Cursor 各論)
- 中盤で合流 (仕様駆動開発 / トレンド論)
- 終盤でVS Codeが合流して最終まとめ

# Antigravity感想

- エージェント管理画面があり、複数エージェントに指示を出しやすい
- 一言で言えば、チャットがしやすいUIって感じ
- まだあんまりGemini3.1 Proらしさは感じたこと無い
- (今回のこのスライド作成させたぐらいしかまだ使っていないけど)

# Antigravityのウワサ

- Google Cloud連携が得意らしい

### CUSTOMIZE GLOBAL SKILLS

#### Skill Custom Paths

In addition to the custom skills folder, Antigravity will search the following paths in order to find skills for the agent.

[+ Add](#)

### INSTALLED MCP SERVERS

[Refresh !\[\]\(341b5bdc31177a6c7da7dc713da0d169\_img.jpg\)](#)

[Open MCP Config](#)

#### No MCP Servers

You currently don't have any MCP Servers installed. Add an MCP server below or add a custom one via the MCP Config.

### ADD MCP SERVERS

#### Cloud Run

Enable Antigravity to deploy apps to Google Cloud Run.

[+ Add](#)

#### Google Kubernetes Engine (OSS)

Enable Antigravity to interact with Google Kubernetes Engine (GKE).

[+ Add](#)

#### Dart

The Dart and Flutter MCP server exposes Dart (and Flutter) development tool actions to compatible AI-assistant clients.

[+ Add](#)

#### Firebase

[+ Add](#)

# Antigravityのウワサ

- 大規模コードベースが強いらしい
- Google Cloud連携が得意らしい
- Google Workspaceと連携すると、自動ドキュメント作成とかメールまとめとかしてくれるらしい
- Googleプラットフォーム向けのAgentとして使っている例が多い
- MCPにGCP連携がたくさんあったので、便利かも？
- 誰か実際に使って教えて欲しい

## ③ Kiro の話

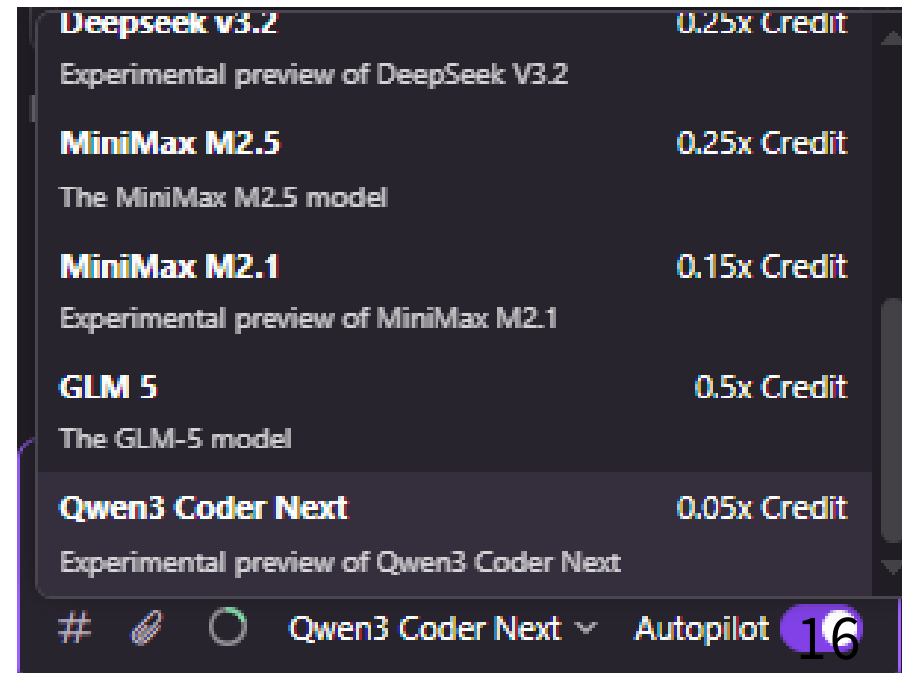
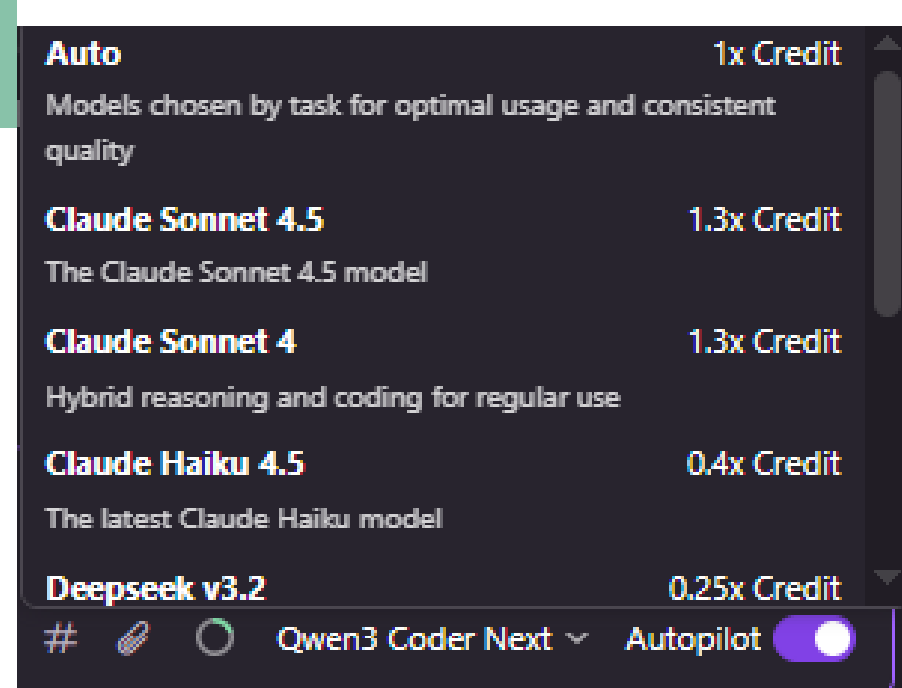
- Kiroの特徴 (1/2, 2/2)
- Kiro使用感
- 無料版の感想 (1/4 ~ 4/4)
- Kiroまとめ

# Kiroの特徴 1/2

- Amazon製
- Spec設計(要件定義、仕様書作成)に重きをおいてる
- BYOKに非対応
- (AmazonとOpenAIが連携したらしいので、GPT系も使えるようになると嬉しいなあ)
- 月額20\$～
- 無料版は50リクエスト/月

# Kiroの特徴 2/2

- 主に中華系のLLMやClaudeが使える
- (おそらく)AWS Bedrock上で動いているLLMが使える



# Kiro使用感

- 要件定義・基本設計・タスクに分解までやってくれる
- `Require.md`, `Design.md`, `Task.md` に分けてくれる

RSS2ZUNDA

- > \_\_pycache\_\_
- > .agent
- > .claude
- > .cursor
- > .github
- > .hypothesis
- ▼ .kiro
  - > settings
  - ▼ specs \ rss2zunda-redesign
    - ≡ .config.kiro U
    - ▼ design.md U
    - ▼ requirements.md U
    - ▼ tasks.md
- > .pytest\_cache
- > .venv
- > output
- ▼ src \ rss2zunda
  - > \_\_pycache\_\_
  - > clients
  - > interfaces
  - > models
  - > modules
  - > music
  - > news
  - > personality
  - > streaming
  - > utils
  - \_\_init\_\_.py M
  - app.py
  - cli.py
  - config.py U
  - exceptions.py U
  - > test\_output

requirements.md U design.md U tasks.md Execution Changes (5 files)

rss2zunda-redesign 1 Requirements > 2 Design > 3 Task list Sync Files Run all tasks

```

1 # Implementation Plan: rss2zunda-redesign
3 ## Overview
   preserving core functionality.
6
7 ## Tasks
8
9 ✓ Task completed | View changes | View execution
10 - [x] 1. Set up project structure and core interfaces
11   - Create directory structure under `src/rss2zunda/`
12   - Create `__init__.py` files for all packages
13   - Define custom exceptions in `exceptions.py`
14   - Requirements: 12.1, 12.2
15
16 ⚡ Start task
17 [ ] 2. Implement data models
18 ✓ Task completed | View changes | View execution
19 - [x] 2.1 Create Pydantic models for Article, AppConfig, and ProcessingResult
20   - Implement `models/article.py` with Article model
21   - Implement `models/config.py` with AppConfig model
22   - Implement `models/processing_result.py` with ProcessingStatus,
23     ArticleResult, ProcessingReport
24   - Requirements: 8.2, 11.1, 10.2, 10.4
25
26 ✓ Task completed | View changes | View execution
27 - [x] 2.2 Write property test for configuration serialization round-trip
28   - **Property 3: Configuration Serialization Round-Trip**
29   - **Validates: Requirements 8.1**
30
31 ✓ Task completed | View changes | View execution
32 - [x] 2.3 Write property test for configuration validation
33   - **Property 4: Configuration Validation Detects Invalid Values**
34   - **Validates: Requirements 8.4**
35
36
37
38
39

```

このコードベースは何をす... +

```

from rss2zunda.utils.filename import

# Test cases from the docstring
print('Test 1:', sanitize_filename('
print('Test 2:', sanitize_filename('
print('Test 3:', sanitize_filename('

# Additional edge cases
print('Test 4:', sanitize_filename('
print('Test 5:', sanitize_filename('
print('Test 6:', sanitize_filename('
print('Test 7:', sanitize_filename('
print('Test 8:', sanitize_filename('
print('Test 9:', sanitize_filename('
"

```

Task: 3.1 Implement filename sanitization utility from tasks.md Status: In Progress

Task: 2.1 Create Pydantic models for Article, AppConfig, and ProcessingResult from tasks.md Status: Completed

Est. Credits Used: 8.01(?)  
Elapsed time: 1h 57m 30s

Kiro

Ask a question or describe a task... ↑

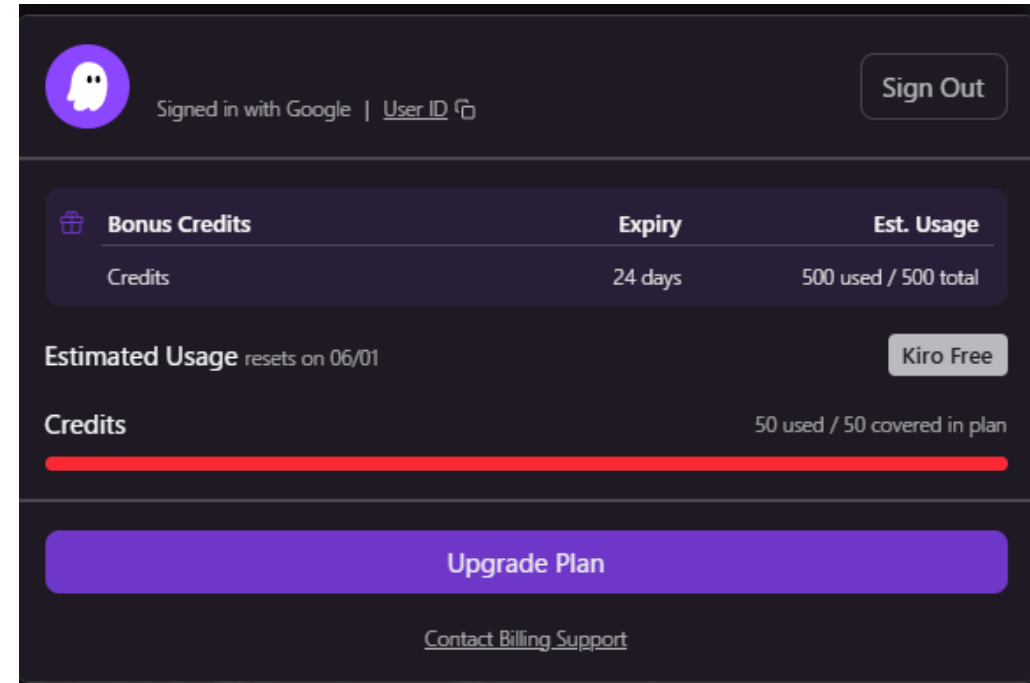
# Qwen3 Coder Next Autopilot


# Kiro使用感


- 要件定義・基本設計・タスクに分解までやってくれる
- `Require.md`, `Design.md`, `Task.md` に分けてくれる
- EARS図やリレーション図やクラス設計まで最初にやってくれる
- `Task.md` では進行状況が見やすいし、Goサインもエディタから出しやすい
- Amazon公式から、AWS向けのSkill(MCP?)を出してるので、AWSとの連携が強い(と思われる)

# 無料版の感想 1/4

- (おそらく)無料枠の対応はキューが後回し
- 日本時間の朝なら早い、夜遅くはキューが混雑してるのか遅い
- スループットも10token/sぐらいで遅いので、ファイルを編集させると30分以上帰ってこない



Signed in with Google | [User ID](#) 

 Bonus Credits	Expiry	Est. Usage
Credits	24 days	500 used / 500 total

Estimated Usage resets on 06/01 Kiro Free

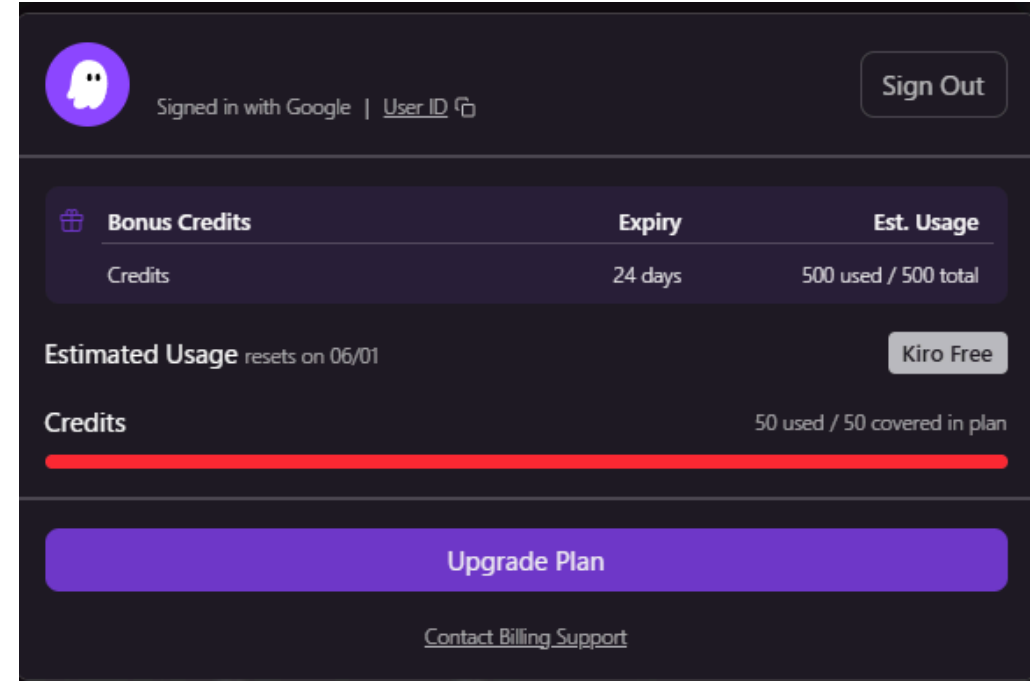
Credits 50 used / 50 covered in plan


[Upgrade Plan](#)

[Contact Billing Support](#)


# 無料版の感想 2/4

- リクエストは結構すぐ消費する
- 有料枠は1000リクエスト/20\$/mo
- 最初の設計だけやるなら、課金しなくても良い気はする
- 無料枠は毎月50リクエストだけど、雀の涙という印象
- 20\$払ってこれだけ?になりそうなので、ちょっと課金を渋ってる



Signed in with Google | [User ID](#) 

[Sign Out](#)

 Bonus Credits	Expiry	Est. Usage
Credits	24 days	500 used / 500 total

Estimated Usage resets on 06/01 [Kiro Free](#)

Credits 50 used / 50 covered in plan

[Upgrade Plan](#)

[Contact Billing Support](#)

# 無料版の感想 3/4

- GLM:Plan + Exec:Qwen3 Coderのパターンにしたらまあまあ持つ
  - 手動でPlanとExecを切り替えようね
- Qwen3の頭脳で足りない場合は、DeepSeekV3でまあまあ解決する
- DeepSeekV3で解決しないときには、泣きながらGLMでなんとかする
- Qwen3 Coder Next : x0.05
- DeepSeek V3.2 : x0.25
- GLM5 : x0.5

# 無料版の感想 4/4

- 昔の書きかけのプロジェクトを完成させようとしたら
  - 600リクエストすぐ消費(初回登録500回、4・5月無料枠50回ずつ)
  - タスク設計の全体の8割しか終わらなかった
  - 細かくテストもしてたのでリクエストがかさむ
  - 残り2割はClaudeCodeにやらせ、Proプランの5h Usageの97%消費
- 正直設計能力だけ欲しいので、BYOK対応してくれると嬉しい

# Kiroまとめ

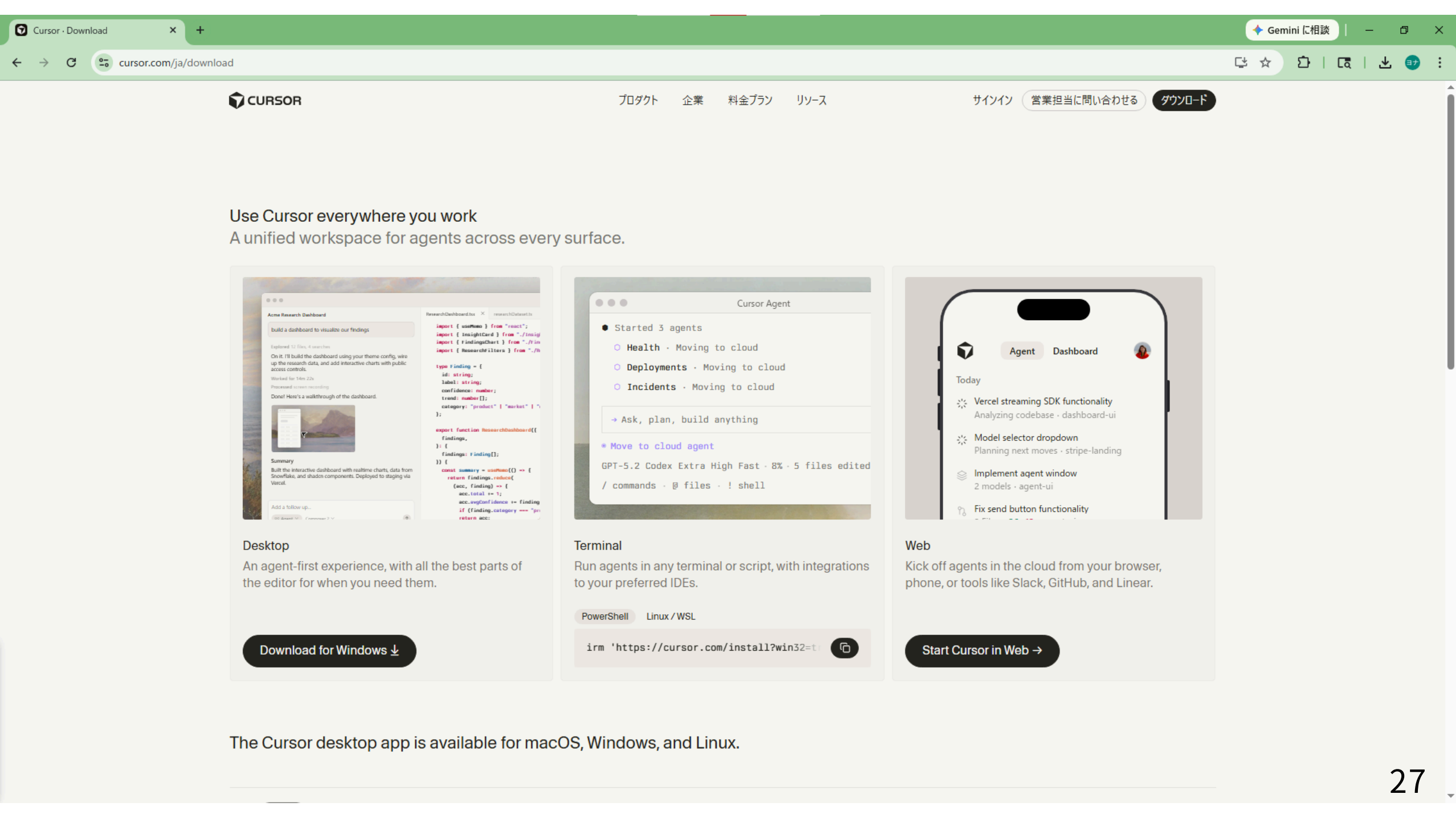
- 押しポイント
  - Spec設計は新たな開発スタイルで面白い
  - UI設計は素晴らしく、.mdファイルが直接タスク管理ダッシュボードになるのは良い
  - 中華系LLMが一通り揃っており、試して遊ぶのにはもってこい
  - 設計用と割り切って、実装は別のIDEに投げるという使い方もできる
- 微妙ポイント
  - 課金額に対しての使用量が渋いなあ
  - 無料枠のスループットが遅い

## ④ Cursorの話

- Cursor (1/3 ~ 3/3)
- Kimi K2.5の性能
- Composer2 (1/3 ~ 3/3)
- Composer2の性能 (要検証)

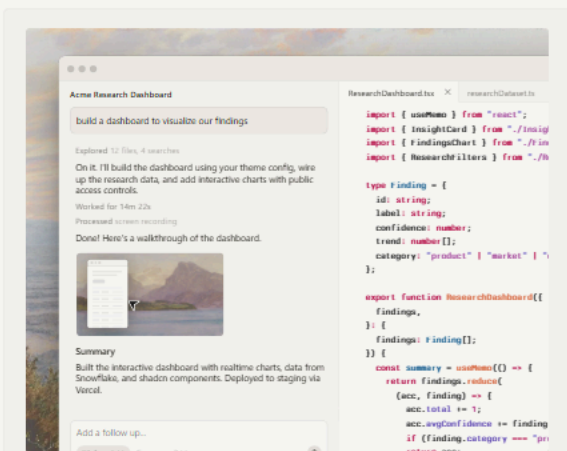
# Cursor 1/3

- Anysphere製



## Use Cursor everywhere you work

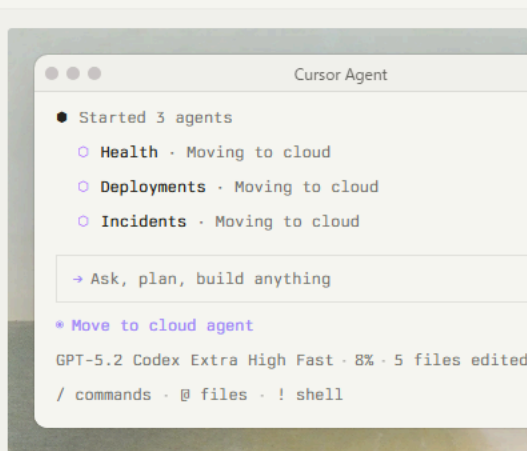
A unified workspace for agents across every surface.



### Desktop

An agent-first experience, with all the best parts of the editor for when you need them.

[Download for Windows ↓](#)

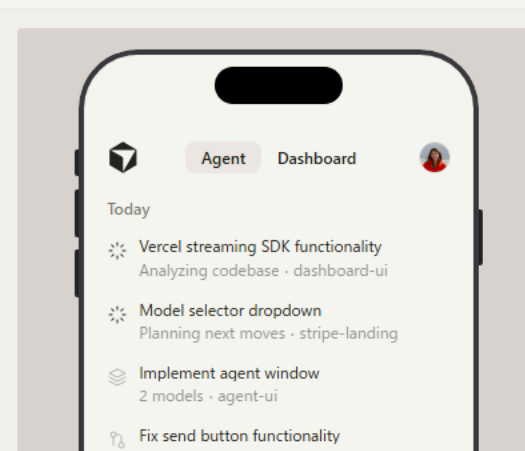


### Terminal

Run agents in any terminal or script, with integrations to your preferred IDEs.

PowerShell Linux / WSL

`irm 'https://cursor.com/install?win32=t'`



### Web

Kick off agents in the cloud from your browser, phone, or tools like Slack, GitHub, and Linear.

[Start Cursor in Web →](#)

The Cursor desktop app is available for macOS, Windows, and Linux.

# Cursor 2/3

- Anysphere製
- トークン消費量改訂系騒動の先駆け(?)
  - 20\$のProプランで500回のリクエスト回数制だったが
  - 2025年6月にトークン使用量制になり、騒がれた
  - (最近どっかで見たな)
- 自社製(?)のLLMであるComposer2が優秀
  - 2026年3月19日にリリースされ、自社製モデルと銘打って登場
  - 中身は Moonshot AIのKimi K2.5のRLじゃねえかと話題に
  - リクエスト枠内の消費が少ないっぽいので、ガンガン使える(?)

# Cursor 3/3

- Composer2のスタイル
  - 要所要所で立ち止まって確認してくれるので、ペアプロ向け
  - モデル自体の自律性よりも、ユーザーの意思を重視するスタイル
  - (ClaudeCodeとかは自律性を高め、長めに自走してくれる、の逆)
- 実際使ってみると割と快適
  - リファクタとか、でっかいコードベースを使わないとか、
  - バイブコーディングでちょっとしたコード書くならこれで良いかも?
  - 個人的、「まず初めにAI powered IDEに課金するなら」でお勧め
- Moonshot Ai製の **Kimi K2.5** 由来の Composer2が結構快適

# Kimi K2.5の性能

- Moonshot AI製の、オープンウェイトなLLM
  - 公式API以外でも、色んなところで使える
- SWE-BenchではGPT-5.3やOpus4.6に迫る勢い (要検証)
- Agent Swarm能力が高い
  - ~100体程度のサブエージェントを役役する能力
- Long Horizon Reasoning
  - 自律性を高めるために、長期的な動向を踏まえてReasoningする能力

# Composer2 1/4

- Cursorが開発したLLM
- モデルIDの名前を覗くと、そこには「kimi」の文字が
- License大丈夫なの?、と騒がれてた
- (国産AIと称しながら中華LLMベースだった話、どっかでも聞いたような……)
- 技術者の良いものは何でも使う精神と、政治はいつの時代も反りが合わない模様

<https://x.com/fynnso/status/2034706304875602030>



Fynn  
@fynnso



英語からの翻訳 原文を表示

CursorでOpenAIのベースURLをいじってたら、これをキャッチした

accounts/anysphere/models/kimi-k2p5-rl-0317-s515-fast

つまりcomposer 2はただのKimi K2.5にRLを施したものの少なくともモデルIDの名前を変えてくれ

この翻訳を評価: 👍 🗨

```

root@api: /opt/debug-sniff
debug-sniff listening on :4180

#1 POST /chat/completions
2026-03-19T18:57:46.159Z

Headers
accept: application/json
accept-encoding: gzip, br
cdn-loop: cloudflare; loops=1
cf-visitor: {"scheme": "https"}
content-length: 30137
content-type: application/json
user-agent: Go-http-client/2.0
x-forwarded-proto: https

Body (30135 bytes)
{
  model: "accounts/anysphere/models/kimi-k2p5-rl-0317-s515-fast"
  messages: [
    {
      role: "system"
      content: "You are an AI coding assistant, powered by Compose..." (2619ch)
    }
    +4 more
  ]
  tools: [
    {
      type: "function"
      function: {
        name: "Shell"
      }
    }
  ]
}

```

Cursor @cursor\_ai · 3月20日

英語からの翻訳

Composer 2 が Cursor で利用可能になりました。

Performance vs. Cost on CursorBench

70% CursorBench score

# Composer2 2/4

- TwitterでMoonshot AI公式からも言及されてた
  - 実はAnysphereとMoonshotAIはパートナシップを結んでいたらしい
  - [https://x.com/Kimi\\_Moonshot/status/2035074972943831491](https://x.com/Kimi_Moonshot/status/2035074972943831491)
  - 補足しておくと、
    - オープンソースベースなのは悪いことじゃない(曰、Cursor開発者)
    - そのベースに選ばれたのは嬉しいこと(曰、MoonshotAI)
    - (米国企業が中華企業のモデルを使ってるという政治云々はさておき)



Kimi.ai 

@Kimi\_Moonshot

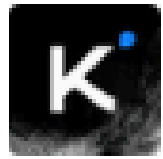


 翻訳を表示 

Congrats to the [@cursor\\_ai](#) team on the launch of Composer 2!

We are proud to see Kimi-k2.5 provide the foundation. Seeing our model integrated effectively through Cursor's continued pretraining & high-compute RL training is the open model ecosystem we love to support.

Note: Cursor accesses Kimi-k2.5 via [@FireworksAI\\_HQ](#) ' hosted RL and inference platform as part of an authorized commercial partnership.



Kimi.ai 

@Kimi\_Moonshot



 英語からの翻訳 [原文を表示](#) 

[@cursor\\_ai](#) チームの Composer 2 のローンチ、おめでとうございます！

Kimi-k2.5 が基盤を提供しているのを見て誇りに思います。Cursor の継続的な事前訓練と高計算リソースの RL 訓練を通じて私たちのモデルが効果的に統合されているのを見るのは、私たちが支援したくてたまらないオープンなモデルエコシステムです。

注：Cursor は、承認された商業パートナーシップの一環として、[@FireworksAI\\_HQ](#) がホストする RL および推論プラットフォーム経由で Kimi-k2.5 にアクセスしています。



Aman Sanger

@amanrsanger

英語からの翻訳 [原文を表示](#)

私たちは、perplexityベースの評価とKimi k2.5で多くのベースモデルを評価しました。Kimi k2.5が最も強力であることが証明されました！

その後、継続的な事前訓練と高計算リソースのRL（4倍のスケールアップ）を行います。

強力なベース、CPTとRL、そしてFireworksの推論とRLサンプラーの組み合わせにより、Composer-2は最先端レベルになります。

ブログで最初からKimiベースに言及しなかったのは見逃しでした。次のモデルではそれを修正します。

この翻訳を評価:



**Kimi.ai** @Kimi\_Moonshot · 3月21日

英語からの翻訳

@cursor\_ai チームの Composer 2 のローンチ、おめでとうございます！

Kimi-k2.5 が基盤を提供しているのを見て誇りに思います。Cursor の継続的な事前訓練と高計算リソースの RL

# Composer2 3/4

- ベースはMoonshot AIのKimi K2.5ではあるものの
  - かなり手間をかけて事後学習をしている様子
  - Tab補完情報のオンライン学習
  - Continued Pretraining(継続学習)
  - Reinforcement Learning(事後強化学習)
  - のためのインフラ整備

# Composer2 4/4

- Cursorがほしかった能力
  - Agent Swarm : 複雑なタスクを分解・サブエージェントに振る能力
  - Long Horizon Reasoning : 長ステップかかる作業も解決できる能力
  - perplexity-base評価の上で最もK2.5が適していると判断した様子
- ベンチマーク結果も公開している
  - <https://cursor.com/resources/Composer2.pdf>
  - 万能で最強というわけではないが、早い安い旨いを満たしてる
  - Cursor IDEの体験も含めて、割と初心者向けの良いモデルでは?という気持ち

Model	CursorBench	SWE-bench Multi.	Terminal-Bench
Composer 2	61.3	73.7	61.7
Composer 1.5	44.2	65.9	47.9
Composer 1	38.0	56.9	40.0
Opus 4.6 High	58.2	75.8 / 77.8	58.0 / 65.4
Opus 4.5 High	48.4	73.8 / 76.2	52.1 / 59.8
GPT-5.4	63.9	76.8 / -	66.5 <sup>†</sup> / 75.1
GPT-5.3 Codex	59.1	74.8 / -	64.8 <sup>†</sup> / 77.3
GPT-5.2	56.5	68.3 / -	60.5 / 62.2
GLM-5	42.7	66.9 / 73.3	59.6 / 56.2
Kimi K2.5	36.0	65.1 / 73.0	47.3 / 50.8

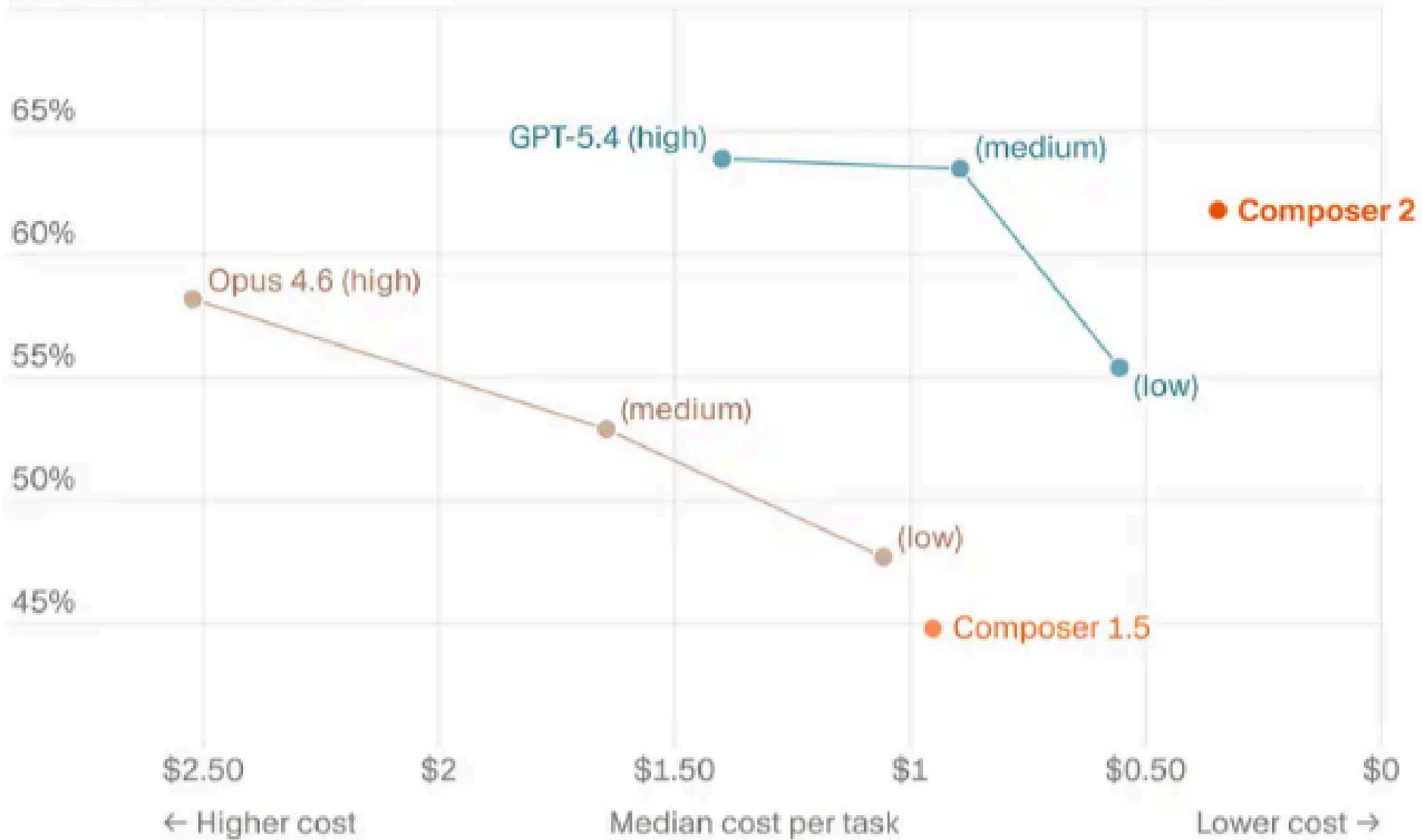
<sup>†</sup>OpenAI safety filters refused 5 GPT-5.4 and 3 GPT-5.3-Codex tasks; refused problems scored as 0.

# 料金一覧 (1Mトークンあたり)

モデル	入力	キャッシュヒット	出力
Composer 2	\$0.5	\$0.2	\$2.5
Opus 4.6	\$5.0	\$0.5	\$25.0
GPT-5.4	\$2.5	\$0.25	\$15.0

## Performance vs. Cost on CursorBench

70% CursorBench score



# Composer2の性能

- **Composer2**
  - Agent Swarm: 規模毎に適宜呼び出し
  - Long Horizon能力: 適宜立ち止まって、指示を仰ぐ
- **Kimi K2.5**
  - Agent Swarm: 最大100体程度
  - Long Horizon能力: 最初の指示だけで細かく分解して作業をやり抜く
- Agent Swarm:(エージェント使役能力)
- Long Horizon能力:(長期的自走能力)解して作業をやり抜く

# Composer2の性能

- 性能真逆じゃない?
  - 私もそう思う
  - Long Horizon
    - ペアプロの性能を上げるために、適宜効くようにしたのかも
  - Agent Swarm
    - (最大使役数は使ってみないとわからない)
- ちなみに
  - Anysphereの技術レポート的には、
  - perplexity-base評価の上で最もK2.5が適していると判断した様子

## ⑤ LLMトレンド論

- 今後のトレンドになりそうな技術
- 結局どのIDEが良いの？
- ハーネスとは
- 最近のLLMのトレンドはハーネス選定

Kiro, Antigravity, Cursorと見てきたけどどれが良いのか

# 今後のトレンドになりそうな技術

- 計画能力と、指示追従性が重視される傾向にあった
- これまで:LLMの単体性能が重要とされていた
- Agent Swarm と Long Horizon Reasoning は今後のトレンドになりそう
- Agent Swarm
  - 長期的な視野を持って、サブエージェントに指示を出して取りまとめる能力
- Long Horizon Reasoning
  - コード量やファイル数が増えても、横断的に編集するための能力
- 単体性能もさることながら、複数体の協調力も求められてるv

# 結局どのIDEが良いの? 1/2

目的ごとに使い分けたほうが良さそう

- 仕様書駆動開発のための仕様をガッツリ作る : **Kiro**
- パパッとモックを作成・Artifact等で確認する : **Antigravity**
- ペアプロして自分の意思・意見強めに編集したい : **Cursor**
- ちょっとAI Powerd IDE触りたい : **Cursor**

# 結局どのIDEが良いの? 2/2

目的ごとに使い分けたほうが良さそう

- どんなLLMを使えるかよりも、どんな設計思想なのかで選んだほうが良さそう
- 選定眼はLLMよりも、**ハーネス**に向けないといけない模様

# ハーネスとは

現実にある道具の名前、安全帯とか馬具とか犬の紐のこと

転じて、AIが好き勝手しすぎないようにある程度  
制御しやすくするためのツールとかの総称

最近の言葉の使い方だと、ラッパー(IDE、Agentツール)もそう呼ばれてる

- ClaudeCode(エージェントツール)
- Cursor(エディタ)
- ある特定の、制御用のAGENT.md

# 最近のLLMのトレンドはハーネス選定

- LLM自体の性能よりも、ハーネスの設計のほうが大事になってきている
- LLMの性能自体がクローズド/オープン共にほぼ横並びになってきている
  - それでもClaude、GPTは頭一つ抜けてはいるが
- よりLLMをうまく使える方が良い結果を生む
  - → Agent.mdやSkill.mdを工夫する必要がある
  - → 設計の良いハーネスのIDEを使うと楽
- (もしかして)
  - 目的に応じてIDE・Agentを使い分ける時代到来か?

## ⑥ VS Code の話

そういえばvscは?

- vscってただのエディタでは?
- LLMと連携する機能ってあった?
- ハーネスってあるの?

## ⑥ VS Code の話

そういえばvscは?

- vscってただのエディタでは?
- LLMと連携する機能ってあった?
- ハーネスってあるの?

その答えが

- Github Copilotでプラン改定があったよね
- どういう理屈?
- VS Codeのこれから

# Github Copilotでプラン改定があったよね

- リクエスト回数方式 → トークン消費量方式
- チャット・指示はトークン数が増える言語は厳しい(日本語は結構デバフを受ける)
- vs code開発チームが対応策を提示してくれてる
  - code ver1.118 のアプデ情報
  - vscの機能として、GithubCopilotのトークン使用量を減らすらしい?
- なんでただのエディタが、一拡張機能に融通きかせてるのか

# どういう理屈？

- vscとcopilotは単なる「エディタと拡張機能」の関係じゃなくなった
- vscが裏側で処理したものをcopilotが受け取る形になった
- ただのモジュールだと思っていたCopilotが、vscと統合した形になる
- (そもそもvscもGithubもMicrosoft製品だし、順当な成長かも?)

# VS Codeのこれから

- vscもこれで晴れて(?) AI(LLM) poweredなIDEの仲間入り
- ベンダーロックインが進むね

.....

# VS Codeのこれから

- vscもこれで晴れて(?) AI(LLM) poweredなIDEの仲間入り
- ベンダーロックインが進むね

(というか、今後はvsc以外のハーネスでCopilot使うと実質デバフでは?)

## ⑦ まとめ

# 結局どのIDEが良いの? 2

- 仕様書駆動開発のための仕様をガッツリ作る : **Kiro**
- パパッとモックを作成して、内蔵ブラウザで確認 : **Antigravity**
- ペアプロして自分の意思・意見強めに編集したい、  
ちょっとAI Powered IDE触りたい : **Cursor**
- Github Copilotの恩恵を最大限受けたい : **VS Code ← NEW**

## 参考文献

- <https://cursor.com/ja/blog/composer-2>
- <https://cursor.com/ja/blog/cursorbench>
- <https://cursor.com/ja/blog/self-summarization>
- <https://cursor.com/ja/blog/composer-2>
- <https://cursor.com/ja/blog/real-time-rl-for-composer>
- <https://cursor.com/ja/blog/composer-2-technical-report>
- <https://x.com/amanrsanger/status/2035079293257359663>
- <https://x.com/fynnso/status/2034706304875602030>
- [https://x.com/Kimi\\_Moonshot/status/2035074972943831491](https://x.com/Kimi_Moonshot/status/2035074972943831491)

本題は以上

そして告知

# 書籍の宣伝

- 新刊「拝啓、AIざかりの君たちへ」
- 技術書典のサイト、もしくは5/10の技書博で購入できます!!!!
- 第十三回技術所同人誌博覧会
  - 2026/5/10(日) 11:00~16:00
  - 大田区産業プラザPiO
- (メロブにはまだ出してないよ!)

## 拝啓、 AIざかりの君たちへ

著者とAIの5年間のについて語る



夜鍋ヨナ 著

# 3周年企画

- みんなのHelloWorld展を開催します
- 本番 6/2 にツアーしましょう

様々なHello Worldの提出ありがとうございました!

現在、ワールド作成中です!!(こあさんありがとうございます!!!!)

各言語の説明を執筆中