2026-06-09時点 AIモデル/ベンダー比較メモ

兎の目線で、2026-06-09時点の主要AIモデルとベンダーの比較メモ。用途別の選択肢やコーディング支援ツールの比較も含む。あくまで現時点での所感であり、今後もしばらくは状況が変わり続けるだろう。

総括

系統	現時点の評価
OpenAI / GPT-5.5 / Codex	複雑な設計判断、レビュー、道筋の整理、ツール利用、ドキュメント生成が強い。人間の知識と経験がまだAI出力より精密な領域でも、補助役としてかなり有益。
Anthropic / Claude Opus 4.8	Agent作業、長い実装修正、コードベース内の反復作業、実装速度で強い。Claude Codeとの相性もよく、現状ではAgent総合力で一歩進んでいる感がある。
Google / Gemini 3.1 Pro / Flash系	巨大コンテキスト、マルチモーダル、Google連携、検索・動画・音声・PDF処理が強い。広い入力を雑に投げる用途では無視し難い。
xAI / Grok	X連携、リアルタイム性、雑談・時事・画像/音声/動画統合の方向。ただし業務・開発用途では信頼性評価がまだ難しい。
Meta / Llama 4系	オープンウェイト、自己ホスト、長文コンテキスト、研究・カスタム用途に強い。最高性能SaaSの代替というより、制御性、研究用途向き。
Mistral系	欧州系、軽量・高速・企業利用・自社統制に魅力。最高難度推論より、実用・統合・コスト制御寄り。
DeepSeek系	コスト性能、推論、コード、数学系で存在感。ただし政治的制約、データ統制、信頼性・供給安定性の評価が必要。
Alibaba / Qwen系	中国Alibaba系の総合モデルファミリ。コード、マルチモーダル、オープンウェイト、低コスト推論の選択肢として有力。ただし政治的制約、データ統制、信頼性・供給安定性の評価が必要。
Microsoft MAI / Copilot内モデル	Microsoft AI。単体モデルというより、開発環境ないし実行環境で統合的に利用可能性のあるAI機能群と見ると良さそう。

主要モデル/ベンダー比較

ベンダー	代表モデル/製品	Pros	Cons	向く用途
OpenAI	GPT-5.5, GPT-5.5 Pro, Codex	複雑問題への対処、設計レビュー、ツール使用、ドキュメント/表計算/スライド生成、プロフェッショナル作業が強い。単なる実装作業よ	重いモデルは高コスト。Agent速度ではClaude系に劣る場面あり。モデル/プランごとの制限も意識が必要。	難問分解、設計判断、コードレビュー、仕様策定、文書化、複雑な調査

ベンダー	代表モデル/製品	Pros	Cons	向く用途
		り、判断の補助で価値が出やすい。		
Anthropic	Claude Opus 4.8, Claude Code	Agent実装、長時間タスク、コードベース修正、作業継続性が強い。手を動かさせるとかなり強い。	深い専門設計で「それらしいが過剰/迂回的」になることがある。枠・従量・プラン制約の管理が必要。	大規模リファクタ、実装Agent、テスト修正、繰り返し編集
Google	Gemini 3.1 Pro, Gemini 3 Flash/Flash-Lite, Gemini Code Assist	巨大コンテキスト、テキスト/画像/動画/音声/PDF入力、検索グラウンディング、関数呼び出し、コード実行など統合力が高い。	会話品質・コード品質はタスクにより揺れる。純粋な開発Agentの安定感はClaude/OpenAIに譲る場面あり。	巨大資料処理、PDF/動画/音声込み調査、Google Workspace連携、低コスト大量処理
xAI	Grok	X連携、リアルタイム時事、音声・画像・動画・コードを含む「全部入り」方向。	安全性・信頼性・業務利用の評価が難しい。企業利用では慎重さが必要。	X文脈、時事雑談、軽い調査、エンタメ寄り生成
Meta	Llama 4 Scout / Maverick	オープンウェイト、自己ホスト、カスタム、長コンテキスト。手元で握れること自体に価値がある。	完全なOSSではなくライセンス制約あり。最高性能SaaSモデルと同等の使い勝手を期待すると厳しい。運用コストも自前。	自己ホスト、社内専用モデル、研究、低レイヤ制御、プライバシー重視
Mistral	Mistral Vibe, Mistral API models	欧州系、企業導入、軽量高速、モデル選択、業務/コードAgent統合。	Frontier最高峰の推論・Agent能力ではOpenAI/Anthropic/Googleに譲る場面あり。日本語や特殊領域は要検証。	企業利用、EU圏要件、軽量処理、コスト制御、API統合
DeepSeek	DeepSeek-V系/R系, Prover系	コスト性能、推論、コード、数学系で強い。安く強いサブモデルとしてはかなり嫌らしい存在。	中国系モデルとして検閲・政治的制約・データガバナンスの懸念が残る。供給安定性や法務評価も必要。	低コスト推論、コード補助、数学/形式証明研究、自己検証用サブモデル
Alibaba	Qwen-Max, Qwen-Coder, Qwen-VL, Qwen-Omni	Alibaba系の総合モデルファミリ。コード、視覚言語、音声/動画を含むマルチモ	DeepSeekと同様、中国系モデルとして検閲・政治的制約・データガバナンス・供給安定性の	低コスト推論、コード補助、自己ホスト/研究、マル

ベンダー	代表モデル/製品	Pros	Cons	向く用途
		ーダル、オープンウェイト展開が広い。DeepSeekと並ぶ中国系モデル比較枠として有用。	評価が必要。日本語業務品質は要検証。	チモーダル実験、DeepSeek比較対象
Microsoft	MAI-Thinking, MAI-Code, Copilot内モデル	GitHub Copilot/VS Code/Visual Studio統合が強い。開発環境に最初から居る、という一点はやはり強い。	単体モデルとしての透明性や外部APIの自由度は限定的。Copilotの課金体系・利用枠に依存。	IDE補完、企業開発環境、GitHub中心ワークフロー

用途別の実務選択

用途	第一候補	第二候補	コメント
複雑な設計判断	GPT-5.5 / Codex	Claude Opus 4.8	「判断の質」を重視するならOpenAI系が堅い場面が多い。
大規模コード修正Agent	Claude Opus 4.8 / Claude Code	Cline + Claude/OpenAI	速度と継続作業はClaude系が強い。
日常補完・IDE統合	GitHub Copilot	Cursor / Continue	モデル性能より統合摩擦の少なさが効く。
巨大PDF/動画/音声/長文	Gemini 3.1 Pro	GPT-5.5 / Qwen-VL系	Geminiの巨大コンテキストとマルチモーダルは強い。Qwen系も実験候補には入る。
コスト重視の大量処理	Gemini Flash系 / DeepSeek / Qwen / Mistral	Llama自己ホスト	精度より単価・速度・量を優先。
自社閉域・自己ホスト	Llama / Mistral / DeepSeek / Qwen	OpenAI互換ローカル基盤	性能より制御性・データ統制が主目的。
画像・動画・音声込みの創作	Gemini / OpenAI / xAI	Qwen / Mistral	目的が制作か調査かで変わる。
X/時事/ネット文脈	Grok	Gemini/Search付きGPT	Grokは強いが、検証なしで事実扱いは危険。

コーディング支援ツール比較

ツール	強み	弱み	評価
GitHub Copilot	IDE/GitHub統合、補完、PR、Issue、複数モデル選択	重量級モデルは課金体系次第で安くない。Git/GitHub連携自体は今や差別化しにくい	日常開発の摩擦削減装置
Cursor	AI IDE体験、エディタ統合、Agent UX	大規模・長時間・最新重量級モデル連打では枠/費用が重い	小〜中規模タスク向け
Claude Code	Claude系Agent性能を最も活かしやすい	Claude系に寄る。枠管理が必要	重いAgent実装向け
Codex	OpenAI系の設計・実装・レビューを統合しやすい	速度/コストは用途次第	難問・レビュー・仕様整理向け
Cline	BYOK、MCP、ローカル/クラウドモデル自由度	UXや安全設計は自分で面倒を見る部分が増える	上級者向けAgent基盤
Continue	設定駆動、モデル/ルール制御、OSS寄り	Copilot/Cursorほど統合体験は丸くない	チーム方針を固定したい場合に有用

所感

2026-06-09時点で単一の勝者はなく、用途や組織の状況に応じて複数のモデルやベンダーを使い分けるのが現実的な選択肢になっている。「これだけ使えばよい」というほど単純な状況ではない。

- Claude Opus系はAgent総合力と速度で一歩進んでいる。特にClaude Code込みなら強い。
- GPT/Codex系は複雑な問題・高度判断で有益。設計判断、反例検出、仕様化では価値が高い。
- Cursorは巨大タスクでは枠・費用面で苦しい。Opus-4.8/GPT-5.5を使うならCopilotの方がメリットが大きい場面もある。
- Copilotはモデル選択制が魅力だが、モデル決め打ちなら魅力半減。Copilotの本質はモデル性能より統合SaaS。
    - 上級者にはGit/GitHub連携は今やMCPやGitHub CLIにより差別化要因にはほとんどならないが、一般組織の水準では導入摩擦の低さとGitHubとの統合性に価値を見いだせる。
- BYOK系Agentが上級者には向く。Cline/Continue + API直契約 + MCP が制御性では強い。特に自社閉域や自己ホストでの利用を考えると、SaaSモデルの枠やプラン制約から解放されるのは大きなメリット。とはいえ、一般組織での導入は運用コストや安全性の観点からハードルが高い。上級者が自分で管理できる範囲で、特定用途に特化したAgentを構築するのには魅力的な選択肢。

用途別1行まとめ

- 中・大規模開発/リファクタリング等のAgent実装は Claude。
- 複雑な設計・専門性の高い判断・高度な問題解決は GPT/Codex。
- 巨大マルチモーダル処理は Gemini。
- 自己ホスト・制御性は Llama / Mistral / DeepSeek / Qwen。
- 日常IDE補完による開発効率の向上は Copilot。
- 上級者向けには BYOK Agent + MCP + 必要なモデルの選択肢もあり。

ただし、どのモデルを使っても、最終判断までAIに預けるのはまだ少しだけ早い。少なくとも高度な設計・安全性・長期保守性が絡む課題では、人間の知識と経験、責任が最後の境界線になる時代。