

# 非エンジニアのための Gemini CLI

## AI でコンテンツ作成を 爆速化しよう！

Google Cloud Japan 合同会社

マーケティング本部

Google Cloud 担当部長

北瀬 公彦

2025/09/16





kkitase

Created by [labs.google/whisk](https://labs.google/whisk)

Google Cloud Japan のマーケティングチームで、  
AI、アプリ開発、インフラといった領域を担当。

日系 Sler、Citrix Systems、IBM、Hortonworks で、  
開発、テスト、プリセールス、サポート、  
マーケティングなど、幅広い業務を経験。

現在は、日本の企業の皆さまが  
AI、クラウドを最大限に活用できるよう支援。

Google Cloud ブログ、技術記事や技術書も多数執筆。

AIツールの利用率は、2月調査時よりも軒並み大きく上昇。

一方でChatGPTやClineなど一部ツールについては公式利用を中止するケースも出てきている。

設問 以下のAIツール・機能・技術について、貴社の開発組織において公式に導入・活用している状況についてお選びください

	今回（2025年7月 調査）		前回の利用率 （2025年2月 調査）	前回からの増減
	使っている	使っていたがやめた		
Gemini	85.5%	0.5%	51.6%	33.9%
GitHub Copilot	82.3%	8.6%	73.4%	8.9%
ChatGPT	73.2%	10.0%	81.4%	-8.2%
Claude	67.7%	1.4%	30.3%	37.4%
NotebookLM	66.8%	2.7%	20.2%	46.6%
Cursor	66.4%	2.7%	31.4%	35.0%
Devin	55.5%	8.6%	20.2%	35.3%
Dify	28.2%	6.4%	17.6%	10.6%
LangChain	20.9%	5.9%	11.7%	9.2%
Perplexity	16.4%	11.8%	11.2%	5.2%
Cline	16.4%	20.0%	6.9%	9.5%

# Agenda

Gemini CLI とは	01
Gemini CLI 基本機能	02
Gemini CLI ベスト プラクティス	03
Gemini CLI でブログを書く	04
Gemini CLI でスライドを作成する	05
Gemini CLI でビデオを作る	06

# Gemini CLI

最先端の AI を

ターミナルに直接統合する

軽量で強力かつアクセスしやすい

オープンソースの CLI ツール

≡ 汎用 AI エージェント



Tips for getting started:

1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. `/help` for more information.

> write a short paragraph about why Gemini CLI is awesome

+ I will start by searching the web for "Gemini CLI" to understand its main features and purpose. Following that, I'll examine the README.md file for a project-specific overview. I will then consult docs/index.md and docs/cli/commands.md to gain a deeper understanding of its capabilities. Finally, I will write a short paragraph summarizing why Gemini CLI is awesome.

~ GoogleSearch Searching the web for: "Gemini CLI features and purpose"

! Uncovering Gemini's Awesome (esc to cancel, 21s)

Using 3 GEMINI.md files

~/code/gemini-cli (release\*)

no sandbox (see /docs)

gemini-2.5-pro (99% context left)

# Gemini CLI の特徴

01

## コードとファイル

コードを生成し、  
ファイルを管理

02

## ツールの呼び出し

他のツールや  
MCP を呼び出し、ローカル開発の管理、テストの実行、  
クラウド サービス連携を実行

03

## 他のアプリとの連携

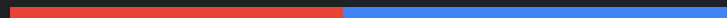
VS Code などと連携し、  
アクション実行やコンテキストを収集が可能

04

## 包括的なコンテキスト

プロジェクトファイル、  
データ、さらにはイメージも含め、最も適切かつ効果的な  
サポートを提供

# Gemini CLI の基本機能



# 設定ファイル: settings.json

認証方式やテーマ、MCP 等の設定を指定

## 階層構造

下の設定が優先

1. デフォルト値
2. ユーザ設定 (~/.gemini/settings.json)
3. プロジェクト設定 (.gemini/settings.json)
4. システム設定  
(/Library/Application Support/GeminiCli/settings.json)
5. 環境変数
6. コマンドライン引数

```
.gemini/settings.json

{
  "theme": "VS2015",
  "sandbox": "docker",
  "toolDiscoveryCommand": "bin/get_tools",
  "toolCallCommand": "bin/call_tool",
  "mcpServers": {
    "mainServer": {
      "command": "bin/mcp_server.py"
    },
    "anotherServer": {
      "command": "node",
      "args": ["mcp_server.js", "--verbose"]
    }
  }
}
```



# Context Files / Memory (GEMINI.md)

- プロジェクト概要、コーディング規約、ガイドラインなどを管理
- @<file\_path> で複数のファイルを import 可能
- /memory show で現在の状況を確認

## 階層構造

- ~/.gemini/[GEMINI.md](#)
- <project\_root>/[GEMINI.md](#)
- <project\_root>/<sub\_dir>/[GEMINI.md](#)

```
.gemini/GEMINI.md

# GEMINI.md

SYSTEM_PROMPT: "あなたは Google Cloud Japan 所属、熟練の IT ライターです。記事の執筆、編集、校正を下記のガイドラインに従って正確に行ってください。"

## スタイルガイド:
- ポジティブなトーンで記述してください。
- Google の公式ブログで使われるような、自然で分かりやすい日本語を使用してください。
- 文法的に正しく、論理的で理解しやすい文章を作成してください。
- 英数字は半角で表記してください。
```

# 基本コマンド

## スラッシュコマンド(!)

- ビルトインの汎用コマンド

```
> /tools
```

/about	バージョン表示
/bug	Github に Issue をレポート
/docs	<a href="#">CLI のドキュメント</a> を開く
/editor	デフォルトエディターの選択
/help	ヘルプの表示
/quit	Gemini CLI を終了
/stats	トークン消費量を確認
/theme	UI のテーマ選択
/tools	利用できるツールを表示

## シェルモード(!)

- シェルの直接操作するためのコマンド

```
> !ls -la
```

```
> !pwd
```

```
> !git status
```

# カスタムスラッシュ コマンド 概要

頻繁に使用するコマンドをGemini CLI 内の  
個人用ショートカットとして保存し、再利用可能

- ファイルの場所
  - グローバル コマンド `~/.gemini/commands/`
  - プロジェクトコマンド `<your-project-root>/.gemini/commands/`
    - コマンド名が同じ場合はプロジェクト優先
- コマンド名
  - `~/.gemini/commands/review/file.toml` -> `/review:file`
  - `~/.gemini/commands/comment.toml` -> `/comment`

```
▼ .gemini
  ▼ commands
    ▼ review
      ⚙ file.toml
      ⚙ prompt.toml
      ⚙ select.toml
      ⚙ comment.toml
      ⚙ whatsnew.toml
```

# カスタムスラッシュ コマンド 作成方法

コマンド定義ファイルは TOML 形式で記述

- フィールド

- `description` (Optional)
  - `/help` メニューに表示される、コマンドの説明
  - 省略した場合は、ファイル名から自動生成
- `prompt` (Required)
  - Geminiモデルに送信するプロンプト本体
  - 単一行、複数行どちらでも記述可能

- 引数の処理

- `{{args}}` でプロンプトの後の入力したテキスト

```
.gemini/git/fix.toml

description = "選択したテキストを校正する"

prompt = """
下記のテキストを、添付のスタイルガイドに従って校正してください。

スタイルガイド: /Users/kkitase/.gemini/style.md

テキスト:
...
{selected_text}
...
"""
```

# MCP サポート

MCP をサポートし、様々なサービスとの連携が可能。

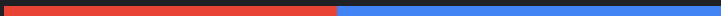
“柴犬が渋谷を旅する、24 秒のショートムービーを作成してください。まずシナリオを考え、次にキーとなる画像をいくつか生成し、それらを基に動画を構成してください。”

```
.gemini/settings.json

{
  "selectedAuthType": "vertex-ai",
  "contextFileName": "GEMINI.md",
  "mcpServers": {
    "veo": {
      "command": "mcp-veo-go",
      "env": {
        "MCP_REQUEST_MAX_TOTAL_TIMEOUT": "240000",
        "MCP_SERVER_REQUEST_TIMEOUT": "30000",
        "GENMEDIA_BUCKET": "vivecoding-genmedia",
        "PROJECT_ID": "vivecoding"
      }
    }
  },
  "preferredEditor": "vscode",
  "theme": "Default Light"
}
```

# Gemini CLI

## ベスト プラクティス



# Context Engineering

Gemini CLI の挙動や精度は、  
コンテキストを適切に管理することで  
向上させることができます。

## コンテキストを提供する主な方法

1. Context Files / Memory (GEMINI.md)
2. Conversations

コンテキストとは、Gemini CLI がユーザーの意図を正確に理解し、的確な応答を生成するために必要とする「背景情報」のことです。

### セッションでのコンテキスト

Context Files  
([GEMINI.md](#))

※ 複数選択可能

+

会話の履歴  
(例: [style.md](#))

※ /compress で時々圧縮可能  
※ @コマンドでファイルやディレクトリ  
を読み込ませることも可能

+

最新の prompt

# 会話履歴

会話履歴はセッションごとに管理され**揮発性**(= /quit すると消える)。

必要に応じて、GEMINI.md やドキュメントに**永続化**して再利用。

## /chat

あるタイミングの会話を保存 / 再開。  
一時的に今のタスクと違うことをしたいときに。

## /clear

画面上と会話履歴をクリア。  
新しいタスクを開始したい場合に。

## At commands (@)

@<path\_to\_file\_or\_directory> と入力し、指定したファイル / ディレクトリをプロンプトに含める。

## /compress

会話を圧縮し、コンテキスト ウィンドウを空ける。



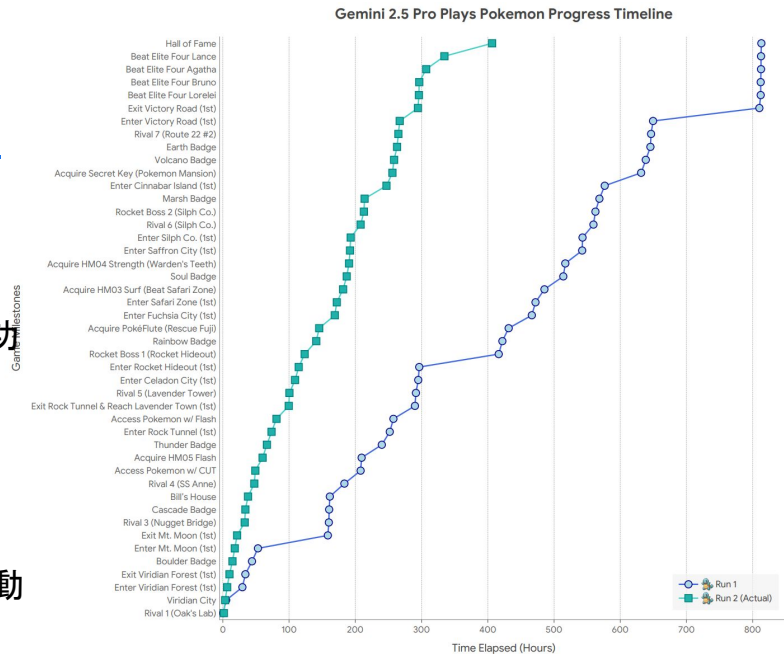
# 参考: コンテキストとの付き合い方

Google DeepMind による

Gemini にポケモン青をプレイさせる動画の考察レポート

[Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities.](#)

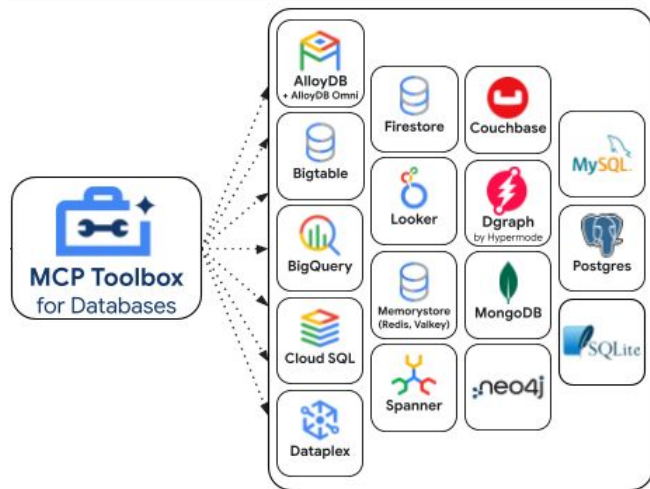
- 10 万トークンが成功の鍵:
  - 10 万トークン程度のコンテキストを処理する能力が成功の礎だった。
- 長すぎるコンテキストの問題:
  - コンテキストが10 万トークンを大幅に超えると、AI は新しい計画を立てるのではなく、過去の膨大な行動履歴の中から行動を繰り返す傾向が見られた。



# MCP Server の活用

MCP サーバーを使用することで Gemini CLI の機能を拡張してデータベース、API、カスタム スクリプト、特殊なワークフローとのやり取りなど、組み込み機能を超えたアクションを実行可能

## MCP Toolbox for Databases



## MCP servers for Genmedia



Vertex AI

Imagen | Veo | Chirp 3 HD | Gemini | Lyria

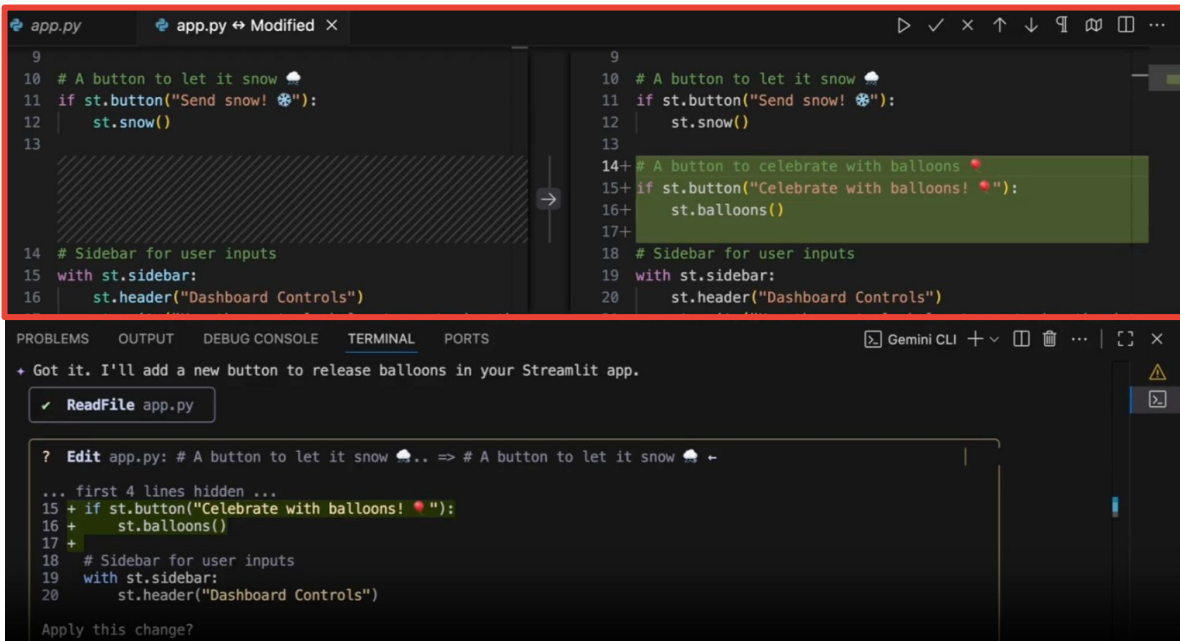
# MCP Server

- Firebase MCP Server
  - [github.com/firebase/firebase-tools/tree/master/src/mcp](https://github.com/firebase/firebase-tools/tree/master/src/mcp)
  - Firebase における Firestore や Firebase Auth など個別サービスの操作が可能
- Google Security Operations and Threat Intelligence MCP Server
  - [github.com/google/mcp-security](https://github.com/google/mcp-security)
  - SecOps、GTI、SCC の情報取得が可能
- VertexMemory - Google Vertex AI / Vector Search MCP Server
  - [github.com/google-octo/google-vertex-vector-search-mcp](https://github.com/google-octo/google-vertex-vector-search-mcp)
  - Vector Search における Memory Object の追加や Search などが可能
- GKE MCP Server
  - [github.com/GoogleCloudPlatform/gke-mcp](https://github.com/GoogleCloudPlatform/gke-mcp)
  - GKE リソースへのアクセス
- その他の MCP Server マーケットプレイス
  - [github.com/punkpeye/awesome-mcp-servers](https://github.com/punkpeye/awesome-mcp-servers)
  - [github.com/modelcontextprotocol/servers](https://github.com/modelcontextprotocol/servers)

# VS Code Integration

- VS Code との接続で、開いているファイルと選択範囲を認識可能
- コード提案を VS Code 内で差分表示でき、差分表示内で承認前にコードを変更可能

## Diff の並列表示



```
9
10 # A button to let it snow ☁
11 if st.button("Send snow! ❄️"):
12     st.snow()
13
14 # Sidebar for user inputs
15 with st.sidebar:
16     st.header("Dashboard Controls")
```

```
9
10 # A button to let it snow ☁
11 if st.button("Send snow! ❄️"):
12     st.snow()
13
14+ # A button to celebrate with balloons 🎈
15+ if st.button("Celebrate with balloons! 🎉"):
16+     st.balloons()
17+
18 # Sidebar for user inputs
19 with st.sidebar:
20     st.header("Dashboard Controls")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

+ Got it. I'll add a new button to release balloons in your Streamlit app.

✓ ReadFile app.py

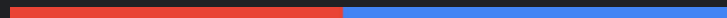
? Edit app.py: # A button to let it snow ☁... => # A button to let it snow ☁ -

```
... first 4 lines hidden ...
15+ if st.button("Celebrate with balloons! 🎉"):
16+     st.balloons()
17+
18 # Sidebar for user inputs
19 with st.sidebar:
20     st.header("Dashboard Controls")
```

Apply this change?

Demo

Gemini CLI で  
ブログを書く

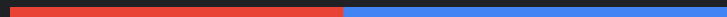


# 手順

- 1 自分の過去のブログや、サンプルになるブログを保存する
- 2 ワークフロー定義ファイル GEMINI.md を作成する
- 3 文体ガイドライン style.md を作成する
- 4 参考になる資料を集める (NotebookLM や Deep Research)
- 5 Gemini CLI に記事を書いてもらう
- 6 人間の手で清書する
- 7 IDE・Gemini CLI を使って効率的に校正する
- 8 ブログに貼るイメージ・動画・GIF を作成する

Demo

Gemini CLI で  
スライドを作成する



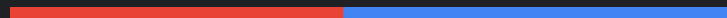
# 手順

- 1 自分の過去のブログや、サンプルになるブログを保存する
- 2 ワークフロー定義ファイル GEMINI.md を作成する
- 3 文体ガイドライン style.md を作成する
- 4 参考になる資料を集める (NotebookLM や Deep Research)
- 5 プロンプトで GAS を作成し、ドラフトを作成する
- 6 ガイドラインに沿った翻訳をする
- 7 Gemini CLI で校正する
- 8 Gemini CLI でスライドノートを作成する



Demo

Gemini CLI で  
ビデオを作る



# AI エージェントによる動画生成

## AI エージェント

茶猫がオーストラ  
リア中を旅する  
30 秒の動画を  
作って



Veo は 8 秒動画しか作成できないので、まず画像を生成し、8 秒動画と、8 秒の音声を 4 本ずつ作成して、最後に合成するプランを作成。

Gemini 2.5

シナリオ 1: xxx  
シナリオ 2: xxx  
シナリオ 3: xxx  
シナリオ 4: xxx

フロントエンド

Google Cloud

Imagen 4  
画像生成

Veo 3  
動画生成

Chirp 3  
音声生成

ffmpeg  
合成

バックエンド

# > GEMINI

Tips for getting started:

1. `/help` for more information.
2. Ask coding questions, edit code or run commands.
3. Be specific for the best results.

Using 2 GEMINI.md files and 4 MCP servers (ctrl+t to view)

YOLO mode (ctrl + y to toggle)

>

~/code/video-studio

no sandbox (see docs)

gemini-2.5-pro (100% context left)

# 手順

- 1 [vertex-ai-creative-studio](#) をクローン
- 2 MCP の設定を行う
- 3 シナリオを作成する
- 4 Gemini CLI にビデオを作成してもらう
- 5 XXX
- 6 XXX

**npm install -g @google/gemini-cli**

```
{ useState, useEffect, useRef } from 'react';
```

```
const WITTY_LOADING_PHRASES = [  
  "Feeling Lucky",  
  "if't panic...",  
  "if't rush perfection (or my code)...",  
  "wrasin' the cogs of the machine...",  
  "and tight, I'm crafting a masterpiece...",  
  "Giving Her all she's got Captain!",  
  "going the distance, I'm going for speed...",  
  "fre got a good feeling about this...",  
];
```

```
const PHRASE_CHANGE_INTERVAL_MS = 15000;
```

```
return () => clearInterval(intervalId);
```

**Thank you!**

