初学者でも今すぐできる、Claude Codeの生産性を10倍上げるTips

Claude Code 初学者 勉強会 2

Oikon

2025/07/05

Who are you?

Oikon

外資IT企業, R&D, 開発

エンジニア歴6年

趣味でツール弄りや個人開発してます

最近はClaude Codeの発信多め（Zenn, 𝕏のハイライト）

𝕏: @gaishi_narou

Claude Code歴

AnthropicとClaude

- Anthropic推し
- Claude 3.5から使用（ちょうど１年くらい）
- Claude Codeは3月に初使用
- Maxプラン開放後愛用（5/1~）

主な使い方:

- ツール作成
- 趣味の開発
- OSSなどコード分析
- LTスライド作成 (new)

今回話すこと

対象: Claude Code初学者

Claude Codeを使ってみたい・使い始めた方

話す内容

X の "CLAUDE CODE 10x productivity workflow" のアレンジ。10分くらいで試せるTips

- 1. IDE(VSCode, Cursor)統合
- 2. Planモード (Shift + Tab 2回)
- 3. CLAUDE.mdのメンテナンス
- 4. `/clear`によるコンテキストの浄化
- 5. `Think`の拡張
- 6. `permissions`の設定
- 7. 最新知識・ドキュメントのInput
- 8. Task (subagent) の活用
- 9. +α: Hooksの活用

1: IDE（VSCode, Cursor）統合

Claude Codeユーザーの多くはIDE統合して使用

CLIツールが苦手な人におすすめ

メリット:

- 変更履歴が見やすい
- 慣れているエディタが使える
- Cursor, GitHub copilotとの併用

Claude CodeはCLIツールとしても活用できるが、
まずはIDEで試すのがおすすめ

個人的には動作が軽いVSCodeを使用している

2: Plan モード (Shift + Tab 2 回)

Claude Codeのタスク実行前にプラン立ててくれる

メリット:

- いきなり走り始めない
- 必要であればプラン修正可能

> ワンショットのプロンプトエンジニアリングは手戻りも多い
>
> コンテキストを大事にする観点でも、Planモードを実行推奨

（余談）settings.json で起動時のデフォルトを Plan モードにできる

```
{
  "permissions": {
    "defaltMode": "plan" // "acceptEdits" | "bypassPermissions"
  }
}
```

3: CLAUDE.md のメンテナンス

CLAUDE.md(CLAUDE.local.md)は、Claude Code起動時に読み込まれるドキュメント

Claude Code の守って欲しいルールを記載する（強制力はないので注意）

ポイント:

- プロジェクトの構造・機能を記載
- 定期的に更新する（`/init`でも可能）
- #で適宜新規の指示を追加

CLAUDE.mdは定期的にメンテナンスすることを推奨。

Anthropic公式のGitHubリポジトリやawesome-claude-codeのCLAUDE.mdを参考にすると良い。

4: /clearによるコンテキストの浄化

Claude Code が期待通りの動作をするためには、**コンテキストウィンドウ**（作業メモリ, 200K）をいかに大事にするかが重要。

意識すること:

- 余計なコンテキストは入れない
- 具体的な指示。余計なファイルを読ませない
- 関係のないタスクは、別のセッションで行う(`/clear`する)

タスクが終了したら定期的に`/clear`をして、コンテキストウィンドウをクリーンにする

（個人的には`/compact`をあまり信用していない。必要ならClaudeにドキュメントに起こしてもらう方がいい）

5: Thinkの拡張

Claude Code には思考トークン予算（Thinking Token Budget）が存在する

Language	31999 tokens	10,000 tokens	4000 tokens
English	**ultrathink**, think harder	**megathink**, think hard	**think**
日本語	深く考えて	よく考えて	考えて

環境変数`MAX_THINKING_TOKENS`の変更可能。常に`ultrathink`したい場合 -> 31999

settings.json (settings.local.json):

```
{
  "env": {
    "MAX_THINKING_TOKENS": "31999" // 1024 ~ 200000
  }
}
```

6: permissionsの設定

permissionsはsettings.jsonで設定

allowとdenyをそれぞれ設定できる。

allowはClaude Codeを使用中に逐次追加できるので、denyの設定をしておくことを推奨

> **注意：必ず守ってくれるという過信は厳禁。**
> **`rm -fr`はすり抜ける報告もあり。**

settings.json (settings.local.json):

```
  "permissions": {
    "allow": [
    ],
    "deny": [
      "Bash(sudo:*)",
      "Bash(rm:*)",
      "Bash(rm -rf:*)",
      "Bash(git push:*)",
      "Bash(git commit:*)",
      "Bash(git reset:*)",
      "Bash(git rebase:*)",
      "Read(.env.*)",
      "Read(id_rsa)",
      "Read(id_ed25519)",
      "Read(**/*token*)",
      "Read(**/*key*)",
      "Write(.env*)",
      "Write(**/secrets/**)",
      "Bash(curl:*)",
      "Bash(wget:*)",
      "Bash(nc:*)",
      "Bash(npm uninstall:*)",
      "Bash(npm remove:*)",
      "Bash(psql:*)",
      "Bash(mysql:*)",
      "Bash(mongod:*)",
      "mcp__supabase__execute_sql"
    ]
  }
}
```

[xより引用]

7: 最新知識・ドキュメントのInput

Claude Codeは最新の知識を持っていないため、追加で知識を与えてあげる必要がある
(Opus/Sonnetは2025年3月までの学習)

知識を追加する方法：

pdf, mdなどドキュメントを直接与える

最新情報をWebSearchで検索してもらう

MCPサーバーを活用する（Context7, Brave-Searchなど）

Context7：代表的なライブラリから最新情報を取得してくれる

導入方法:

```
claude mcp add context7 -s project -- npx -y @upstash/context7-mcp
```

-s : スコープ（user, project, local）

8: Task (subagent) の活用

Claude Codeの**Task**は、Subagentが実行している

- 軽量
- 並列起動可能
- subagentは独自のContext Windowを持つ
- 単独のタスクで動作し完了すると解放される
- 親agentで使用可能なツール（MCPサーバーなど）を使える

使い方：

"use (multiple) subagents ..."

簡単なタスクは積極的にSubagentに任せることがおすすめ

```
● Task(Summarize CHANGELOG.md)
  └ Done (1 tool use · 18.4k tokens · 35.9s)

● Task(Summarize CLAUDE.md)
  └ Done (1 tool use · 19.6k tokens · 28.8s)

● Task(Summarize README.md)
  └ Done (1 tool use · 18.4k tokens · 23s)

● Task(Summarize RELEASE_NOTES.md)
  └ Done (1 tool use · 16.3k tokens · 21.8s)

● Task(Summarize TODO.md)
  └ Done (1 tool use · 15.8k tokens · 21.1s)

● Task(Summarize sample.md)
  └ Done (1 tool use · 23.4k tokens · 24.7s)

● Task(Summarize template.md)
  └ Done (1 tool use · 15.4k tokens · 22.5s)

● Update Todos
  └ ☒ Find all markdown files in the root directory
    ☒ Read and summarize each markdown file using subagents

✶ Processing… (85s · ↓ 948 tokens · esc to interrupt)

> 
```

+α: Hooks の活用

7月1日に追加された新機能！

Claude Codeのアクションを検知して、事前に決められた動作を、
指定のタイミングで行う機能。

導入のメリット:

- 必ず実行してくれる（=ルールを守らせる）
- コンテキストサイズの縮小
- 拡張性の向上

すぐ導入できる例：タスク完了時の通知オンを鳴らす

/Hooksを実行してNotificationかStopに以下のコマンドを入れる

```
afplay /System/Library/Sounds/Sosumi.aiff
```

shell scriptの実行もでき、拡張性がかなり高い！

```
Select hook event:
  1. PreToolUse - Before tool execution
  2. PostToolUse - After tool execution
❯ 3. Notification - When notifications are sent
  4. Stop - Right before Claude concludes its response
  5. SubagentStop - Right before a subagent (Task tool call) concludes its response

  Enter to acknowledge risks and continue · Esc to exit
```

さらに使いこなすためのキーワードたち

- Slash Command
- カスタム Slash Command
- MCP サーバーの追加
- ヘッドレスモード(claude -p "prompt")
- Git Worktree
- `--dangerously-skip-permissions`
- CodeRabbit + `/pr-comments`
- 音声入力（Aqua Voice）
- 著名なエンジニアによるコンテキストの明示（t_wada, Kent Beck, Fowler...）
- ccusage

まとめ

- Claude CodeはCLIツールゆえにできることが多い（というか多すぎる）
- 全部の機能を熟知する必要はない。まずは使える機能を試すと良い
- Pro, Maxプランなら定額なので、色々試しまくる！
- いい活用例があったら積極的に共有してください！

おすすめ資料

- Claudeの紹介 - Anthropic
- Claude Code Best Practices \ Anthropic
- How I Use Claude Code | Philipp Spiess - spiess.dev
- Claude Code: Best Practices and Pro Tips - htdocs.dev
- ClaudeLog: Claude Code Docs, Guides & Best Practices
- Claude Code を初めて使う人向けの実践ガイド - Zenn
- Claude Code 逆引きコマンド事典 - Zenn
- Claude Code Deep Dive "
- Claude Code にコマンド一発で MCP サーバを簡単設定 - Zenn
- awesome-claude-code