Claude Code Best Practices

Anthropic - Claude Code best practices | Code w/ Claude

https://www.youtube.com/watch?v=gv0WHhKelSE

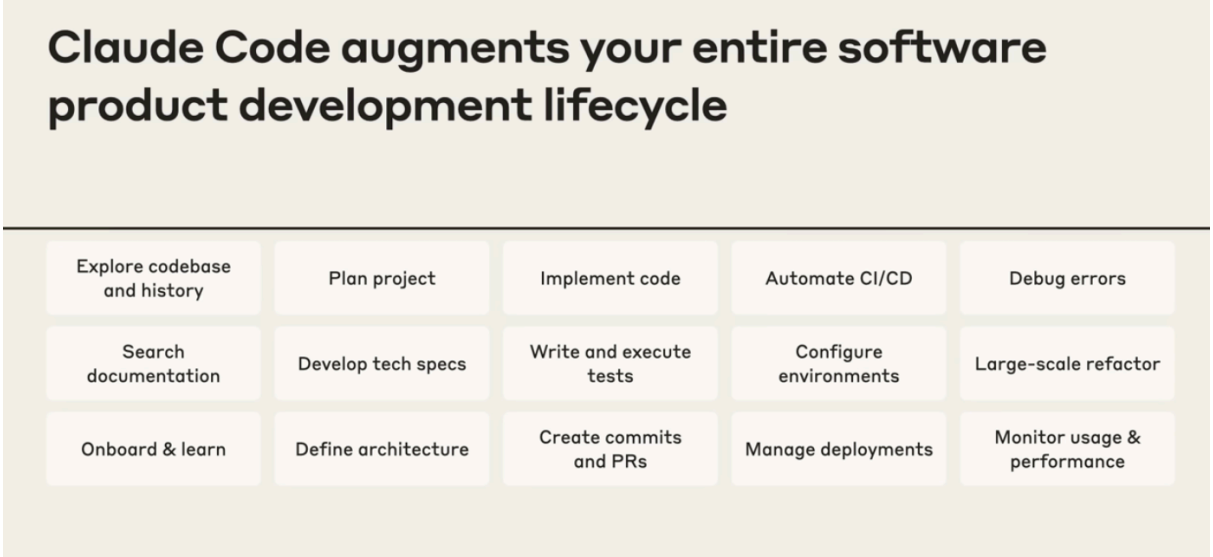
Edits & creates files, uses your CLI and MCP tools, and creates commits. Tackles both narrow tasks like bug fixes and broad refactors. (Powerful actions & integrations)

Maintains awareness of your project structure. No need to manually add files to context - Claude agentically searches your codebase as needed. (Codebase awareness)

Claude shows its work, ande uses tiered permissions system that balances power with control(Transparency)

Queries go directly to Anthropic models via API without intermediate servers. Supports Anthropic APIs or APIs on Amazon Bedrock or Google Cloud Vertex AI

Use Cases



- Good to learn code base fast and easy
- Gain good knowledge about the project like new changes, git history etc
- Thought Partner (Sugerir e pedir opiniões)

What can Claude do?

1. Discover
2. Design
3. Build
4. Deploy
5. Support & Scale

Best Practices

1. Use CLAUDE.md files!
    a. Create these in strategic locations to document commands, style guidelines, and key context.
    b. Put it on a project, home directory...

    c. Information like: “This is how you should run this test” or “Here is our style guide”
2. Permission Management
    a. Curate allowed tools to reduce interruptions during coding sessions.
3. Integration Setup
    a. Install tools like gh CLI for seamless GitHub workflows, and let Claude handle version control.
4. Context Management
    a. Use /clear regularly to maintain focused conversations.

Effective workflows

- Planning and TODOs - First ask Claude to read relevant files and make a detailed plan, verify claudes to do list.
- Smart vibe coding - Leverage test driven development and regular commits to vibe code with the right level of guardrails
- Use screenshots to guide & debug - Use screenshots to compare implementations against mockups, and provide guidance to Claude on how to design

Advanced Techniques

- Multi-Claude & Parallelization - Use separate instances for coding, reviewing, and testing.
    - Deploy work-trees of sub-agents to tackle complex problems simultaneously.
- Use escape - Balance letting Claude work and interjecting at the right times. Press escape once to stop. Press escape twice to jump back in the conversation.

Tool expansion & MCP

- Integrate bash tools, MCP servers, and custom slash commands. Note: Claude is both an MCP client & server!

Um **server MCP** pode ter alguns significados diferentes dependendo do contexto, mas geralmente se refere a um **servidor que implementa ou utiliza o protocolo MCP (Model Context Protocol)**.

O **MCP** é um protocolo aberto, lançado pela OpenAI em 2024, que serve para conectar **modelos de linguagem** (como eu) a **ferramentas externas, bancos de dados, APIs e outros serviços**.

👉 Nesse contexto:

- Um **MCP server** é um programa que **expõe recursos ou funcionalidades** (como dados, documentos, funções de uma API, etc.) de forma padronizada para que o modelo de linguagem consiga acessá-los.
- O **MCP client** (geralmente o modelo de linguagem ou uma aplicação que o hospeda) se conecta a esse servidor para **consultar dados, executar ações ou receber informações em tempo real**.

Um exemplo simples:

- Você pode ter um MCP server que conecta a um banco de dados de vendas.
- O modelo, atuando como cliente MCP, pode perguntar: *"Qual foi o faturamento do último mês?"*.
- O server responde com os dados já formatados.

Headless automation

- Use -p mode for CI/CD pipelines and large-scale changes