M02 Context Management

Managing the context window for efficient sessions

Methodology Beginner

The 4 Context Zones

Zone	Threshold	Required action
Green	0–75%	Normal work, no action needed
Yellow	75–85%	Suggest `/compact`
Orange	85–90%	Compact without waiting
Red	90%+	Urgent, `/clear` if needed

Context Sources (by priority)

```
1. ~/.claude/CLAUDE.md       Global rules
2. ./CLAUDE.md               Project rules (root)
3. subfolder/CLAUDE.md       Local rules
(cumulative)
4. @file                     Explicit inline
import
5. --context flag            File passed via CLI
```

Management Commands

```
/compact        # Summarize + free ~40% of
context
/clear          # Full reset (loses history)
/cost           # View tokens used / remaining
/status         # Detailed context state
```

Adding Context Intelligently

```
# In the prompt
@src/auth.ts Look at this function

# Via CLAUDE.md (persistent)
Always read package.json before working on deps.

# Via --context (session)
claude --context "We use TypeScript strict mode"
```

Strategies by Scenario

Long session (refactor) — Make regular checkpoints with `/compact` every 30 min. Save important decisions in `CLAUDE.md`.

Complex debugging — Provide only the relevant files with `@file`. Avoid loading the entire project at once.

Multi-file — Use agents (`--agent`) to delegate: each sub-agent has its own isolated context.

What Consumes the Most

Source	Impact
Large files read in full	Very high
Conversation history	High
Verbose command outputs	High
Overly long CLAUDE.md	Medium
Long Claude responses	Medium

Optimizing CLAUDE.md

```
# Good: short and actionable rules
- Always use TypeScript strict mode
- Prefer pnpm over npm

# Bad: long narrative context
We have been working on this project since 2023
and the team decided after many debates to...
```

Common Anti-patterns

Do not paste large JSON or raw logs into the prompt — use a temporary file and `@file` instead. Do not ignore context warnings: at 85%+, response quality degrades noticeably.