T03 # Permission Modes

Technical Beginner

Tool access control --- from safest to most permissive

Available Modes

Mode	Flag	Recommended use
Default	(none)	Daily development
Auto-accept edits	`Shift+Tab`	Code reviews
Auto-accept all	`Shift+Tab x2`	Known repetitive tasks
Full bypass	`--dangerously-skip-permissions`	Headless CI/CD

Tool Whitelist

```
# Allow only specific tools
claude --allowedTools "Read,Grep,Glob"

# Block specific tools
claude --disallowedTools "Bash,Write"

# Useful combinations
claude --allowedTools "Read,Edit,Bash(git*)"
```

Configuration in settings.json

```
{
  "permissions": {
    "allow": [
      "Bash(git log*)",
      "Bash(npm test*)",
      "Read",
      "Edit"
    ],
    "deny": [
      "Bash(rm*)",
      "Bash(sudo*)"
    ]
  }
}
```

Permission Hierarchy

Permissions accumulate and are inherited in this order:

1. `~/.claude/settings.json` — global user
2. `.claude/settings.json` — project (shared)
3. `.claude/settings.local.json` — project (local, gitignored)
4. CLI flags — session only

Glob Patterns for Bash

```
# Allow git only
"Bash(git *)"

# Allow npm test and build
"Bash(npm test*)", "Bash(npm run build*)"

# Allow file reading
"Bash(cat *)", "Bash(ls *)"
```

Best Practices

CI/CD — Always use `--dangerously-skip-permissions` with a sandboxed environment (Docker, ephemeral container). Never on a shared production machine.

Sensitive projects — Restrict Bash tools with precise globs in `.claude/settings.json`. Commit this file so the whole team uses the same constraints.

Audit — Claude's actions are logged in `~/.claude/logs/`. Verifiable at any time.