/sp-requesting-code-review

Source: `~/.claude/skills/sp-requesting-code-review/SKILL.md`

---

name: requesting-code-review description: Use when completing tasks, implementing major features, or before merging to verify work meets requirements

Requesting Code Review

Dispatch superpowers:code-reviewer subagent to catch issues before they cascade.

Core principle: Review early, review often.

When to Request Review

Mandatory:

- After each task in subagent-driven development
- After completing major feature
- Before merge to main

Optional but valuable:

- When stuck (fresh perspective)
- Before refactoring (baseline check)
- After fixing complex bug

How to Request

1. Get git SHAs:

```
BASE_SHA=$(git rev-parse HEAD~1)  # or origin/main
HEAD_SHA=$(git rev-parse HEAD)
```

2. Dispatch code-reviewer subagent:

Use Task tool with superpowers:code-reviewer type, fill template at `code-reviewer.md`

Placeholders:

- `{WHAT_WAS_IMPLEMENTED}` - What you just built
- `{PLAN_OR_REQUIREMENTS}` - What it should do
- `{BASE_SHA}` - Starting commit
- `{HEAD_SHA}` - Ending commit
- `{DESCRIPTION}` - Brief summary

3. Act on feedback:

- Fix Critical issues immediately
- Fix Important issues before proceeding
- Note Minor issues for later
- Push back if reviewer is wrong (with reasoning)

Example

```
[Just completed Task 2: Add verification function]

You: Let me request code review before proceeding.

BASE_SHA=$(git log --oneline | grep "Task 1" | head -1 | awk '{print $1}')
HEAD_SHA=$(git rev-parse HEAD)

[Dispatch superpowers:code-reviewer subagent]
```

```
  WHAT_WAS_IMPLEMENTED: Verification and repair functions for conversation index
  PLAN_OR_REQUIREMENTS: Task 2 from docs/plans/deployment-plan.md
  BASE_SHA: a7981ec
  HEAD_SHA: 3df7661
  DESCRIPTION: Added verifyIndex() and repairIndex() with 4 issue types

[Subagent returns]:
  Strengths: Clean architecture, real tests
  Issues:
    Important: Missing progress indicators
    Minor: Magic number (100) for reporting interval
  Assessment: Ready to proceed

You: [Fix progress indicators]
[Continue to Task 3]
```

Integration with Workflows

Subagent-Driven Development:

- Review after EACH task
- Catch issues before they compound
- Fix before moving to next task

Executing Plans:

- Review after each batch (3 tasks)
- Get feedback, apply, continue

Ad-Hoc Development:

- Review before merge
- Review when stuck

Red Flags

Never:

- Skip review because "it's simple"
- Ignore Critical issues
- Proceed with unfixed Important issues

- Argue with valid technical feedback

If reviewer wrong:

- Push back with technical reasoning
- Show code/tests that prove it works
- Request clarification

See template at: requesting-code-review/code-reviewer.md

Revision #5
Created 2026-02-18 08:39:55 UTC by John
Updated 2026-06-21 20:00:58 UTC by John