/sentry-skill-creator

Source: `~/.claude/skills/sentry-skill-creator/SKILL.md`

---

name: skill-creator description: Create new agent skills following the Agent Skills specification. Use when asked to "create a skill", "add a new skill", "write a skill", "make a skill", "build a skill", or scaffold a new skill with SKILL.md. Guides through requirements, writing, registration, and verification.

Create a New Skill

Guide the user through creating a new agent skill following the Agent Skills specification. Follow each step in order.

Step 1: Understand the Skill

Gather requirements before writing anything.

Ask the user:

1. What should this skill do? (one sentence)
2. When should an agent use it? (trigger phrases)
3. What tools does the skill need? (Read, Grep, Glob, Bash, Task, WebFetch, etc.)
4. Where should the skill live? (which plugin or directory)

Determine the skill name:

- Lowercase alphanumeric with hyphens, 1-64 characters
- Descriptive and unique among existing skills
- Check the target skills directory to avoid name collisions

Choose a complexity tier:

Tier	Structure	Use When
Simple	`SKILL.md` only	Self-contained instructions under ~200 lines
With references	`SKILL.md` + `references/`	Domain knowledge that agents load conditionally
With scripts	`SKILL.md` + `scripts/`	Workflow automation needing Python scripts
Full	All of the above	Complex skills with automation and domain knowledge

Read `${CLAUDE_SKILL_ROOT}/references/design-principles.md` for guidance on keeping skills focused and concise.

Step 2: Study Existing Skills

Before writing, study 1-2 existing skills that match the chosen tier. Look for skills in the target repository or plugin to understand local conventions.

Read `${CLAUDE_SKILL_ROOT}/references/skill-patterns.md` for concrete examples of each tier.

Also read `CLAUDE.md` (or `AGENTS.md`) at the repository root for repo-specific conventions that the skill should follow.

Step 3: Write the SKILL.md

Create `<skill-directory>/<name>/SKILL.md`.

Frontmatter

The YAML frontmatter **must** be the first thing in the file. No comments or blank lines before `---`.

```
---
name: <skill-name>
description: <what it does>. Use when <trigger phrases>. <key capabilities>.
---
```

Required fields:

- `name` — must match the directory name exactly
- `description` — up to 1024 chars; include trigger keywords that help agents match user intent

Optional fields:

- `model` — override model (`sonnet`, `opus`, `haiku`); omit to use the user's default
- `allowed-tools` — space-delimited list (e.g., `Read Grep Glob Bash Task`); omit to allow all tools
- `license` — license name or path (add when vendoring external content)

Body Guidelines

Write the body in **imperative voice** — these are instructions, not documentation.

Do	Don't
"Read the file and extract..."	"This skill reads the file and extracts..."
"Report only HIGH confidence findings"	"The agent should report only HIGH confidence findings"
"Ask the user which option to use"	"You may want to ask the user..."

Structure:

1. Start with a one-line summary of what the skill does
2. Organize steps with `## Step N: Title` headings
3. Use tables for decision logic and mappings
4. Include concrete examples of expected output
5. End with validation criteria or exit conditions

Size limits:

- Keep SKILL.md under **500 lines**
- If approaching the limit, move reference material to `references/` files

- Load reference files conditionally based on context (not all at once)

Attribution

If the skill is based on or adapted from external sources, add an HTML comment **after** the frontmatter closing `---`:

```
---
name: example
description: ...
---

<!--
Based on [Original Name] by [Author/Org]:
https://github.com/example/original-source
-->
```

Step 4: Create Supporting Files

References (`references/`)

Use for domain knowledge the agent loads conditionally.

```
<name>/
├── SKILL.md
└── references/
    ├── topic-a.md
    └── topic-b.md
```

Reference from SKILL.md with:

```
Read `${CLAUDE_SKILL_ROOT}/references/topic-a.md` for details on [topic].
```

Keep each reference file focused on one topic. Use markdown with tables and code blocks.

Scripts (`scripts/`)

Use for workflow automation that benefits from structured Python.

```
<name>/
├── SKILL.md
└── scripts/
    └── do_thing.py
```

Script requirements:

- Always use `uv run` to execute: `uv run ${CLAUDE_SKILL_ROOT}/scripts/do_thing.py`
- Add PEP 723 inline metadata for dependencies:

```
# /// script
# requires-python = ">=3.12"
# dependencies = ["requests"]
# ///
```

- Output structured JSON for agent consumption
- Run from the **repository root**, not the skill directory
- Document the script's interface in SKILL.md (arguments, output format)

Assets (`assets/`)

Use for static files the skill references (templates, configs, etc.).

LICENSE

Include a LICENSE file in the skill directory when vendoring content with specific licensing requirements.

Step 5: Register the Skill

Registration steps vary by repository. Check the repository's `CLAUDE.md` or `README.md` for specific instructions.

1. **Verify directory-name match** — confirm the directory name matches the `name` field in SKILL.md frontmatter exactly
2. **Update documentation** — add the skill to any skills index or table in README.md
3. **Update permissions** — if the repo has `.claude/settings.json`, add `Skill(<plugin>:<name>)` to the `permissions.allow` array
4. **Check CLAUDE.md** — read the repository's `CLAUDE.md` for any additional registration steps specific to that project

Step 6: Verify

Run through this checklist before finishing:

Frontmatter

- ☐ `name` matches directory name
- ☐ `description` is under 1024 characters
- ☐ `description` includes trigger keywords
- ☐ No content before the opening `---`

Content

- ☐ SKILL.md is under 500 lines
- ☐ Written in imperative voice
- ☐ Steps are numbered and clear
- ☐ Examples of expected output included
- ☐ Reference files loaded conditionally (not unconditionally)

Registration

- ☐ Directory name matches frontmatter `name`
- ☐ Skill added to repo documentation (README or equivalent)
- ☐ Permissions updated (if applicable)
- ☐ Any repo-specific registration steps completed (check CLAUDE.md)

Scripts (if applicable)

- ☐ Uses `uv run ${CLAUDE_SKILL_ROOT}/scripts/...`
- ☐ Has PEP 723 inline metadata
- ☐ Outputs structured JSON
- ☐ Documented in SKILL.md

Report any issues found and fix them before completing.

Revision #4
Created 2026-02-18 08:40:01 UTC by John
Updated 2026-05-31 20:01:35 UTC by John