Agent Skills 原理 + 应用 + 落地

1. Skills 定义

核心定义

来源：Claude 官方

Skills 是一组指令、资源和脚本的文件夹，模型可调用，可组合，可移植，可扩展。

官方文档：Claude 官方文档 - 扩展模型能力概览 / Claude 官方文档 - 开始使用 Agent Skills，文档明确定义，Skills 是指令、脚本、资源的文件夹，模型可动态加载以提高在特定任务上的表现。

Skills 是模型可调用的文件夹，包含指令、脚本和资源，模型根据任务上下文决定是否调用，无需手动选择。Claude 会扫描可用 Skills 的元数据，仅在需要时加载完整内容，保持上下文效率。

目录结构规范

```
skill-name/
├── SKILL.md          # 必须，入口文件
├── scripts/          # 可选，可执行脚本
│   ├── main.py
│   ├── utils.sh
│   └── ...
├── references/       # 可选，参考文档
│   ├── guide.md
│   ├── examples.md
│   └── ...
└── assets/           # 可选，模板、资源文件
    ├── template.xlsx
    ├── logo.svg
    └── ...
```

SKILL.md 文件格式

```
---
name: skill-name
description: 简短描述，决定何时调用
---

# 指令内容

Markdown 格式的指令，模型会遵循这些指令
```

与 Prompt 的核心差异

	Prompt	Skill
持久性	一次性，每次重新输入	持久，跨会话可用
加载方式	手动，每次粘贴	按需自动加载
上下文效率	占用固定上下文	渐进式加载，高效
可组合性	难以组合多个	可与其他 Skills 组合
可移植性	依赖对话	文件夹，跨平台可用
可扩展性	受限于单次输入长度	可添加脚本、资源、参考文档

Skills 的核心优势

- **上下文效率高**：Claude 启动时仅加载所有 Skills 的元数据（名称和描述），约 100 tokens，而非全部内容。只有当任务匹配时，才加载完整的 SKILL.md 指令，保持上下文窗口高效利用。
- **可组合性强**：多个 Skills 可以同时存在，模型会自动判断哪些需要调用，并按顺序加载。例如：处理 PDF 的 Skill + 生成 Excel 的 Skill + 数据分析 Skill，可组合成一个工作流。
- **可移植性**：Skills 是纯文本文件，可在 Claude 平台、Claude Code、API 调用中使用，一次创建，处处可用。
- **可扩展性**：可以添加脚本、模板、参考文档，赋予模型更多能力。脚本可以执行确定性操作，节省 token 和提高稳定性。

官方定义

来源：Claude 官方文档 - 扩展模型能力概览

> 模型调用工具时，上下文是关键。Skills 是模型可调用的文件夹，包含指令、脚本和资源，模型会根据任务上下文决定是否调用。Skills 可以携带大量上下文内容，保持文件夹内容高效组织，模型只在需要时通过文件系统读取相关内容。

与 MCP 的关系

> MCP 为模型提供连接外部服务的能力，而 Skills 教模型如何使用这些连接。MCP 是工具层，Skills 是知识层。

Skills 的适用场景

- 重复性任务：每次需要解释相同的规则、格式、流程
- 组织特定工作流程：团队的代码规范、文档模板、审查清单
- 处理特定文件格式：Excel 公式规则、PDF 处理流程、数据分析方法
- 领域知识：公司政策、合规要求、最佳实践
- 跨平台一致性：在 Claude 平台、Claude Code、API 中保持相同行为

参考资料

- 官方文档：Claude 官方文档 - 扩展模型能力概览
- 官方文档：Claude 官方文档 - 开始使用 Agent Skills
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践
- 开放标准：agentskills.io

2. Prompt / Tool / Skill / MCP 区别

核心定义

来源：Claude 官方文档 - Claude 如何使用 Prompt 工程

四个概念

概念	定义	比喻	特性
Prompt	一次性指令	口头说一句话	每次重新输入，不持久
Tool	模型可调用的函数	手机上的 App	结构化输入输出，有限能力
Skill	可复用的指令集合	给新员工的工作手册	渐进式加载，高效组合
MCP	连接外部服务的协议	手机的蓝牙/WiFi	标准化，跨平台

关系图

MCP 可以 Skill

Skill 文件夹 = 工具集合的指南，MCP 是 Skill 的 Resources 或脚本调用对象。

四个核心区别

区别 1：可复用性

- Prompt：一次性，每次重新输入
- Tool：模型可调用，但能力有限
- Skill：可复用的指令集合，渐进式加载，高效组合
- MCP：标准化协议，跨平台，可连接外部服务

区别 2：上下文效率

- Prompt：每次占用固定上下文
- Tool：结构化输入输出，有限能力
- Skill：渐进式加载，只在需要时加载完整内容
- MCP：标准化，跨平台，可连接外部服务

区别 3：可组合性

- Prompt：难以组合多个
- Tool：可与其他 Tool 组合
- Skill：可与其他 Skills 组合，模型自动判断哪些需要调用
- MCP：可连接多个 MCP 服务器，模型可调用

区别 4：适用场景

- Prompt：临时指令，不重复的任务
- Tool：需要结构化输入输出的操作
- Skill：重复性任务，组织特定工作流程，处理特定文件格式，领域知识，跨平台一致性
- MCP：连接外部服务，访问数据库，调用 API，跨平台集成

官方定义

来源：Claude 官方文档 - 扩展模型能力概览

> Skills 是模型可调用的文件夹，包含指令、脚本和资源，模型会根据任务上下文决定是否调用。Skills 可以携带大量上下文内容，保持文件夹内容高效组织，模型只在需要时通过文件系统读取相关内容。

参考资料

- 官方文档：Claude 官方文档 - 扩展模型能力概览
- 官方文档：Claude 官方文档 - 开始使用 Agent Skills
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践
- 开放标准：agentskills.io

3. 核心原理——渐进加载

来源：Claude 官方文档 - 扩展模型能力概览，Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践

三层加载模型

为什么这么"省着用"？

原因 1：上下文是稀缺资源

- 模型的上下文窗口有限
- 每个 token 都有机会成本
- 加载不相关内容会挤占其他信息的空间

原因 2：模型自己会判断

- 启动时只加载元数据（~100 tokens/Skill）
- 模型根据任务上下文决定是否调用
- 不需要手动选择，模型会智能匹配

原因 3：渐进式设计

- 第一层：~100 tokens/Skill（元数据）
- 第二层：< 5k tokens（SKILL.md 指令）
- 第三层：无限制（引用的文件、脚本执行结果）

对比传统方案

	传统方案	Skills 方案
启动时	加载所有内容	只加载元数据
调用时	加载相关内容	加载 SKILL.md 指令
执行时	加载更多内容	按需加载引用文件
上下文效率	低，内容堆积	高，渐进式加载

官方定义

来源：Claude 官方文档 - 扩展模型能力概览

> 模型调用工具时，上下文是关键。Skills 是模型可调用的文件夹，包含指令、脚本和资源，模型会根据任务上下文决定是否调用。Skills 可以携带大量上下文内容，保持文件夹内容高效组织，模型只在需要时通过文件系统读取相关内容。

参考资料

- 官方文档：Claude 官方文档 - 扩展模型能力概览
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践

4. Skill 的最小结构——SKILL.md 怎么写

来源：Claude 官方文档 - 开始使用 Agent Skills

最小结构

一个 Skill 只需要一个文件夹和一个 SKILL.md 文件。

```
skill-name/
└── SKILL.md
```

最小 SKILL.md 模板

```
---
name: skill-name
description: 简短描述，决定何时调用
---

# 指令内容

Markdown 格式的指令，模型会遵循这些指令
```

SKILL.md 的两部分

第一部分：YAML frontmatter（元数据）

```
---
name: skill-name
description: 简短描述，决定何时调用
---
```

必填字段

- `name`：技能名称（小写字母、数字、连字符，最多 64 字符）
- `description`：简短描述，决定何时调用

可选字段

- `allowed-tools`：限制可使用的工具（如 `Read, Write, Bash`）
- `model`：指定模型（如 `claude-sonnet-4-20250514`）
- `context`：设置上下文（`fork` 表示在子代理中运行）
- `agent`：指定代理类型（`Explore`、`Plan`、`general-purpose` 或自定义代理名）
- `hooks`：定义生命周期钩子（`PreToolUse`、`PostToolUse`、`Stop`）
- `user-invocable`：是否在 slash 菜单中显示（默认 `true`）
- `disable-model-invocation`：是否禁止模型调用（默认 `false`）

第二部分：Markdown 指令（正文）

```
# 指令内容

Markdown 格式的指令，模型会遵循这些指令

## 步骤

1. 第一步：读取输入文件
2. 第二步：处理数据
3. 第三步：生成输出

## 规则

- 规则 1：不要修改原文件
- 规则 2：保持格式一致

## 示例

输入：xxx
输出：yyy

## 模板

使用以下模板生成报告：
...

## 参考资料

- 参考文档：guide.md
- 脚本：scripts/main.py
```

YAML 字段汇总

字段	必填	说明
`name`	推荐	技能名称（小写字母、数字、连字符，最多 64 字符）
`description`	推荐	简短描述，决定何时调用
`allowed-tools`	可选	限制可使用的工具（如 `Read, Write, Bash`）
`model`	可选	指定模型（如 `claude-sonnet-4-20250514`）
`context`	可选	设置上下文（`fork` 表示在子代理中运行）
`agent`	可选	指定代理类型（`Explore`、`Plan`、`general-purpose` 或自定义代理名）
`hooks`	可选	定义生命周期钩子（`PreToolUse`、`PostToolUse`、`Stop`）
`user-invocable`	可选	是否在 slash 菜单中显示（默认 `true`）
`disable-model-invocation`	可选	是否禁止模型调用（默认 `false`）

description 写作公式

来源：Claude 官方文档最佳实践 - 编写有效的 Skills

> 始终用第三人称写 description。description 会注入系统提示，视角不统一会导致发现问题。

官方 Skill Creator 的建议

> 模型知道它们的能力，重点是提供模型不具备的信息：程序性知识、组织特定的工作流程、领域知识、可执行的脚本。

参考资料

- 官方文档：Claude 官方文档 - 开始使用 Agent Skills
- 官方文档：Claude 官方文档 - 编写有效的 Skills
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践
- 开放标准：agentskills.io

5. 最佳实践——把 Skill 当"产品"打磨

来源：Claude 官方文档 - 编写有效的 Skills，Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践

三条黄金规则

编写原则

- 简洁优先：模型已经很聪明，只需要它不知道的东西
- 自由度匹配：高自由度用指导，低自由度用脚本
- 渐进式加载：元数据 ~100 tokens，指令 < 5k tokens（推荐），资源按需读取
- 第三人称描述：description 会注入系统提示，视角不统一会导致发现问题
- 测试多模型：Haiku 够用吗？Sonnet 清晰高效吗？Opus 不要过度解释吗？

工作流程

- 评估驱动：先创建评估，再编写 Skill
- 迭代开发：与 Claude 一起编写和测试 Skill
- 观察思考：注意 Claude 如何使用 Skill，调整内容和结构

参考资料

- 官方文档：Claude 官方文档 - 编写有效的 Skills
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践
- 开放标准：agentskills.io

6. Skill + MCP——组合成生产级 Agent

来源：Claude 官方文档 - 扩展模型能力概览，Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践，Claude 官方文档 - 开始使用 Agent Skills

概念原理

工具和 Skill 工作原理

工具	MCP 提供能力	工具	Skill 教使用方法
数据库	连接数据库，执行查询	数据分析	分析工作流程，报告模板
文件系统	读写文件，搜索文件	代码生成	代码规范，项目结构
浏览器	导航，截图，点击	测试	测试策略，断言模式
API	调用 REST API	文档	文档模板，风格指南

组合模式

- 工具提供能力，Skill 教如何使用
- 工具连接外部服务，Skill 教工作流程
- 工具执行操作，Skill 教最佳实践
- 工具访问数据，Skill 教领域知识
- 组合使用：Claude 智能调用工具，根据任务上下文决定调用哪个 Skill，按需加载 MCP 工具，执行 Skill 指导的工作流程，生产级 Agent

参考资料

- 官方文档：Claude 官方文档 - 扩展模型能力概览
- 官方博客：Claude 官方博客 - 为真实世界部署 Agent Skills 的工程实践
- 官方文档：Claude 官方文档 - 开始使用 Agent Skills
- 开放标准：agentskills.io

AI通识学习

学习平台：https://www.bilibili.com/video/BV1ZNPCzMFTZ/?share_source=copy_web

学习资料：https://www.kdocs.cn/l/ckqbnTcqbTdd?linkname=ai

本教程是在Claude官方文档、官方博客、Claude Code、Agent Skills、MCP、Prompt Engineering等一手资料基础上，结合实践经验整理的通用知识。

以下内容来自官方文档，是Skills生态的核心设计理念：

- 渐进式加载：元数据 ~100 tokens，指令 < 5k tokens（推荐），资源按需读取
- 模型自主判断：模型根据任务上下文决定是否调用，无需手动选择
- 跨平台一致性：Claude 平台、Claude Code、API 中保持相同行为
- 上下文效率优先：启动时仅加载元数据，只在需要时加载完整内容

Agent Skills 是开放标准，规范最初开发，最新更新于 2025 年 12 月。目前已被 OpenAI、Microsoft、Cursor、GitHub、JetBrains 等主流 AI 工具采用，正在成为跨平台 AI 能力扩展的通用格式。

参考资料：https://agentskills.io/home#adoption-and-ecosystem

学习平台：https://www.bilibili.com/video/BV1ZNPCzMFTZ/?share_source=copy_web

本文由 Claude 生成，已通过多个工具交叉验证事实准确性。

B站/公众号/YouTube：唐国梁Tommy

关注微信公众号
获取更多学习资源

官方网址：TG.Tomy.com 限时享上线

版权所有 Tony Tommy，仅供个人学习使用，请勿商用或传播。