A PRACTICAL HANDBOOK · 2026 EDITION

สร้าง Claude Skill แบบไม่ต้องรู้โค้ด

คู่มือ prompt-first สำหรับครีเอเตอร์ คนทำการตลาด และฟรีแลนซ์ ที่อยากให้ AI ทำงานซ้ำได้แทนตัวเอง

PROMPT-FIRST | ภาษาไทยทั้งเล่ม | 12 บท + ภาคผนวก | OFFICIAL-SOURCE ONLY

Prompt-First Series

Vol. 01 — ภาษาไทยทั้งเล่ม · 12 บท + ภาคผนวก

01

TABLE OF CONTENTS · สารบัญ

สารบัญ

หนังสือเล่มนี้ออกแบบให้อ่านเรียงทีละบทก็ได้ หรือกระโดดไปบท Skill ตัวอย่างที่ใกล้กับงานคุณก่อน ทุกบทมี prompt พร้อมใช้และตัวอย่าง output จริง

ส่วนที่ 1 · เข้าใจก่อนลงมือ · บทนำ–บทที่ 4

ส่วนที่ 2 · 6 SKILL ตัวอย่างใช้ได้จริง · บทที่ 5–10

TABLE OF CONTENTS · สารบัญ

สารบัญ (ต่อ)

ส่วนที่ 2 (ต่อ) · ส่วนที่ 3 · ภาคผนวก

ส่วนที่ 2 (ต่อ) · บทที่ 8–10

ส่วนที่ 3 · ทดสอบและแพ็ก · บทที่ 11–12

ภาคผนวก

ส่วนที่ 1

I

เข้าใจก่อนลงมือ

ก่อนจะลงมือสร้าง Skill เรามาเข้าใจกันก่อนว่า Skill คืออะไร ทำไมคนไม่รู้โค้ดถึงสร้างเองได้ ต้องเตรียมเครื่องอย่างไร และจะคิด Skill ที่ดีได้อย่างไร

บทนำ · CHAPTER ZERO

00

Skill คืออะไร และทำไม คนไม่รู้โค้ด ก็สร้างเองได้

ลองนึกถึง **คู่มือเล่มเล็ก** ที่บอกให้พนักงานคนหนึ่ง ทำงานซ้ำได้ทุกครั้ง ในแบบที่คุณอยากให้เป็น — Skill คือสิ่งนั้น แต่คนที่ทำงานให้คุณคือ Claude

บทนำนี้จะอธิบายว่า Skill ต่างจาก prompt ธรรมดาอย่างไร ทำไมมันเหมาะกับงานที่คุณทำซ้ำ ๆ และมีตัวอย่าง Skill ที่ครีเอเตอร์ทั่วไปลองทำได้ในวันเดียว

- ✕ ไม่ใช่หนังสือสอนเขียนโค้ด — ใช้การคุยกับ Claude เป็นหลัก
- ✕ ไม่ใช่ตำราอ้างอิง API — เน้นทำของใช้ได้จริง
- ✕ ไม่ใช่ทฤษฎี — ทุกบทมี Skill ตัวอย่างที่ลงมือทำตามได้

Skill ต่างจาก prompt ธรรมดายังไง

Prompt คือคำสั่งหนึ่งครั้ง Skill คือ **คู่มือถาวร** ที่ Claude หยิบมาใช้เอง เมื่อเจองานแบบเดียวกัน คุณไม่ต้องพิมพ์ คำสั่งซ้ำทุกครั้ง และทีมคุณก็ได้ผลลัพธ์เหมือนกันทุกคน

01

Prompt ธรรมดา

พิมพ์ใหม่ทุกครั้ง · คุณภาพขึ้นกับว่าจำได้ดีแค่ไหน · ส่งต่อให้ทีมยาก เพราะอยู่ในหัวคุณ · ปรับ tone ต้องเริ่มใหม่

02

Skill

เก็บเป็นไฟล์ ในเครื่อง · Claude หยิบใช้เอง · ทีมใช้ได้พร้อมกัน · ปรับปรุงทีละนิดได้ · เวอร์ชันใหม่ไม่กระทบของเก่า

🎯 **ตัวอย่างเทียบให้เห็นภาพ**

สมมติว่าคุณต้องเขียนแคปชั่นโพสต์ขายของทุกวัน — **prompt ธรรมดา** คือคุณพิมพ์ใหม่ทุกครั้ง และหวังว่าจะจำ รายละเอียด tone ของแบรนด์ได้ครบ **Skill** คือคุณบอก Claude ครั้งเดียวว่าแบรนด์คุณพูดยังไง แล้วทุกครั้งที่ขอแคปชั่น Claude จะใช้ tone นั้นให้อัตโนมัติ

Skill เหมาะกับใคร

- **ครีเอเตอร์** ที่โพสต์คอนเทนต์ทุกวันแล้วอยากให้ tone นิ่ง
- **คนทำการตลาด** ที่มี workflow รีวิว / launch ซ้ำทุกเดือน
- **ฟรีแลนซ์** ที่ส่งงานลูกค้าหลายคนและอยากให้คุณภาพไม่ตก
- **เจ้าของธุรกิจ** ที่อยากเขียน SOP ให้ทีมใช้ตามได้

Skill ต่างจาก plugin / connector / MCP ยังไง

ในระบบของ Claude มีอะไรหลายอย่างที่ฟังดูคล้ายกัน แต่ทำหน้าที่ต่าง — ส่วนนี้สรุปสั้น ๆ พอให้เห็นภาพ ไม่ต้องจำหมดก็ได้

Plugin / Connector

ช่องทางให้ Claude เชื่อมกับบริการภายนอก เช่น Gmail, Drive · จัดการโดย Anthropic หรือ partner

MCP Server

บริการกลางที่เปิดให้ Claude ใช้เครื่องมือเฉพาะทาง · เหมาะกับงานที่ต้องเชื่อมหลายระบบ

Skill

คู่มือ + เครื่องมือย่อยที่คุณเขียนเอง · ทำงานในเครื่องคุณ · ใช้ใน Claude Code ได้ทันที

💡 **เลือกอะไรเมื่อไหร่**

ถ้าต้องการ **เชื่อมบริการภายนอก** → plugin/connector ถ้าต้องการ **workflow ของตัวเองที่ใช้ซ้ำ** → Skill ถ้าต้องการ **ระบบกลางให้หลายทีมใช้** → MCP server
หนังสือเล่มนี้เน้นที่ Skill เพราะเริ่มต้นง่ายที่สุด และเป็นจุดเริ่มของการ systemize งานส่วนตัว

ทำไม Skill เหมาะกับคนที่มี workflow ซ้ำ ๆ

ทุกครั้งที่คุณคิดว่า **“ฉันทำงานนี้ทุกสัปดาห์ น่าจะมีวิธีให้เร็วกว่านี้”** นั่นคือสัญญาณว่างานนั้นเป็น Skill ได้แล้ว Skill เปลี่ยน workflow ในหัวคุณ ให้เป็น **กระบวนการที่ Claude ทำตามได้** และคุณตรวจอีกที

ตัวอย่าง Skill ที่นึกออกทันที

ก่อนเริ่มอ่านต่อ ลองดู 5 ตัวอย่างนี้ ถ้ามีอันไหนที่คุณเคยอยากทำ นั่นคือ Skill แรกที่ควรลองสร้างหลังอ่านบทที่ 4 จบ

📝

สรุป meeting

ส่ง transcript / note เข้าไป ได้ summary + decisions + action items + follow-up message

🎬

YouTube → caption pack

ส่ง URL วิดีโอ ได้ caption ภาษาไทย + summary + Facebook post 3 แบบ + Reels caption

📣

Facebook post pack

ใส่ข้อมูลสินค้า ได้ launch post + founder story + FAQ + objection handling

✅

Brand voice rewriter

ส่งข้อความเข้าไป ได้เวอร์ชันที่ตรง tone ของแบรนด์คุณ พร้อมเหตุผลว่าปรับอะไร

🧩

ตัวอย่างที่ห้าที่ขายดีที่สุด

Checklist ก่อนส่งงานลูกค้า — รับ deliverable แล้วเช็คก่อนส่ง ว่าครบไหม spelling โอเคไหม ตรงตาม brief ไหม มีอะไรเสี่ยงไหม freelancer หลายคนใช้แล้วลด rework ได้ครึ่งหนึ่ง

ทั้ง 5 ตัวอย่างมีจุดร่วมเดียวกัน

- งาน **แคบและชัด** — รู้ว่า input อะไร output อะไร
- **ทำซ้ำได้** — คุณทำงานแบบนี้อย่างน้อยสัปดาห์ละครั้ง
- **มี standard ในหัว** — คุณรู้ว่า output ที่ดีหน้าตาเป็นยังไง

Skill ที่ดีคือ workflow ชัด ไม่ใช่คำสั่งกว้าง ๆ

ผู้เริ่มต้นส่วนใหญ่ทำพลาดในจุดเดียวกัน — สร้าง Skill ที่กว้างเกินไป เช่น “ช่วยทำการตลาดให้หน่อย” ผลที่ได้คือ Claude เดาว่าคุณต้องการอะไร และเดาผิดบ่อย

✕

Skill ที่กว้างเกิน

“ช่วยทำ marketing ทุกอย่าง” · “เขียน content ให้หน่อย” · “ดูแลโซเชียลให้” — ไม่มี trigger ชัด · output ไม่แน่นอน

✓

Skill ที่ชัด

“แปลง long-form video เป็น Facebook post pack” · “สรุป meeting เป็น action items + email” — รู้ว่าใช้เมื่อไหร่และให้อะไร

⚠ **สัญญาณว่า Skill กว้างเกิน**

- คุณอธิบายไม่ได้ใน 1 ประโยคว่า Skill นี้ทำอะไร
- ไม่แน่ใจว่าจะใช้กับ input แบบไหน
- output ไม่มี format ตายตัว
- คุณเองยังไม่รู้ว่า “ดี” คืออะไร

ในบทที่ 4 จะมีสูตรชื่อ **Skill Brief 7 ช่อง** ที่ช่วยให้คุณเปลี่ยนไอเดียกว้าง ๆ เป็น Skill ที่ชัดพอจะลงมือสร้างได้ ตอนนี้แค่จำว่า **Skill = workflow ชัด ไม่ใช่ความหวัง**

ลองให้ Claude อธิบาย Skill ให้คุณฟังก่อน

ก่อนอ่านต่อ ลอง copy prompt ด้านล่างนี้ไปวางใน Claude แล้วดูว่ามันอธิบาย ได้ตรงกับที่คุณเข้าใจไหม ถ้าตรง — เยี่ยม คุณพร้อมไปบทที่ 1 แล้ว ถ้ายังงง — ก็ปกติ เพราะหนังสือทั้งเล่มนี้คือคำตอบ

prompt · ให้ Claude อธิบาย Skill แบบคนทั่วไป

```
> ช่วยอธิบายให้ฉันฟังหน่อยว่า Claude Skill คืออะไร
โดยอธิบายแบบที่คุยกับ คนที่ใช้ AI เป็นแต่ไม่รู้โค้ด

กติกา:
- ภาษาไทย ใช้คำง่าย ๆ ไม่ใช้ศัพท์เทคนิค
- ยกตัวอย่างจากงานของครีเอเตอร์หรือคนทำเพจ
- ห้ามยาวเกิน 6 บรรทัด
- ปิดท้ายด้วยคำถามว่าฉันอยากเริ่มทำ Skill อะไรก่อน
```

KEY TAKEAWAY

Skill ไม่ใช่เวทมนตร์ — มันคือ **คู่มือสั้น ๆ** ที่บอก Claude ว่าเมื่อเจองานแบบนี้ ต้องทำอะไร ทำตามขั้นไหน และห้ามทำอะไร ถ้าคุณเขียนคู่มือให้คนได้ คุณก็เขียน Skill ได้

→ บทต่อไป
บทที่ 01 · เตรียมเครื่องแบบไม่ต้องเป็นสายโค้ด
พลิกหน้า →

บทที่ หนึ่ง · CHAPTER ONE

01

เตรียมเครื่องแบบ ไม่ต้องเป็นสายโค้ด

คนไม่รู้โค้ดส่วนใหญ่ติดที่ขั้นแรก — “ต้องลง terminal อะไรบ้าง?” บทนี้ตอบทั้งหมดในหน้าเดียว แล้วให้คุณเปิด Claude Code ใช้งานได้ ภายใน 15 นาที โดยไม่ต้องเข้าใจระบบลึก

หลักการคือ **ติดตั้งเฉพาะที่จำเป็น** และ **ให้ Claude ตรวจให้ก่อนรันคำสั่งใด ๆ**

- × ไม่ใช่บทสอน Linux / shell — copy/paste เป็นพอ
- × ไม่ต้องเข้าใจ npm, pip, brew ลึก — แค่รู้ว่าเป็น “ตัวติดตั้ง”
- × ไม่ต้องตั้ง config ระบบเพิ่ม — Claude Code ทำให้แทบทั้งหมด

ต้องมีอะไรบ้างก่อนเริ่ม

3 อย่างที่ต้องมีก่อนติดตั้ง Claude Code ไม่ต้องเข้าใจรายละเอียด แค่รู้ว่าแต่ละอันคืออะไรก็พอ

01

Claude Account

บัญชี Claude ที่ใช้ Claude Code ได้ — ถ้าใช้ claude.ai อยู่แล้ว ใช้บัญชีเดียวกันได้เลย

02

Terminal

ช่องดำ ๆ ในเครื่องสำหรับคุยกับเครื่อง · macOS = Terminal app · Windows = PowerShell

03

Folder

โฟลเดอร์สำหรับเก็บโปรเจกต์ Skill — สร้างใหม่ที่ Desktop ได้ ตั้งชื่ออะไรก็ได้

💡 **ทำความเข้าใจ Terminal ครั้งเดียวจบ**

Terminal คือ **ช่องคุยกับเครื่อง** ที่ใช้พิมพ์คำสั่งแทน การคลิก ทั้งบทนี้คุณจะพิมพ์แค่ 3-4 คำสั่งเท่านั้น และทุกคำสั่งจะ **copy/paste ได้** ไม่ต้องจำ
กฎเดียวที่ต้องจำ — **ถ้าไม่เข้าใจ ให้ถาม Claude ก่อนรัน**

วิธีคิดเรื่อง CLI สำหรับคนไม่รู้โค้ด

- CLI = Command Line Interface = พิมพ์คำสั่งแทนการคลิก
- ทุกคำสั่งใน terminal **copy ได้** — ไม่ต้องจำ
- Enter = รันคำสั่ง · Ctrl+C = หยุดคำสั่งที่กำลังรัน
- ไม่เข้าใจคำสั่ง **อย่ารัน** — ให้ Claude อธิบายก่อน

ติดตั้ง Claude Code ด้วย official install link

Anthropic มี official install command ที่อัปเดตอยู่เสมอ อยู่ที่ **claude.com/code** เปิดหน้านี้แล้ว copy คำสั่ง install ที่ตรงกับเครื่องคุณ — macOS / Linux ใช้อันเดียวกัน Windows มีอันแยก

> ⚠️ **ห้ามรันคำสั่ง install จากแหล่งอื่น**
>
> มีเว็บปลอมที่ลอกหน้า claude.com/code เปลี่ยน install URL แล้ว ฝัง malware เช็คให้ดีก่อนรันว่า URL คือ **claude.com** เท่านั้น ไม่ใช่ claude.ai-something.com

หลังติดตั้งเสร็จ ตรวจว่าใช้ได้จริง

ตรวจเวอร์ชัน

```
$ claude --version
```

ถ้าเห็นเลขเวอร์ชัน (เช่น `2.x.x`) = ติดตั้งสำเร็จ ถ้าเห็น **command not found** = ยังไม่สำเร็จ ให้ปิด terminal เปิดใหม่ก่อน ถ้ายังไม่ได้ ค่อยถาม Claude ช่วยตรวจ (มี prompt ให้ในหน้าถัดไป)

เปิด Claude Code ในโฟลเดอร์แรก

Claude Code ทำงานในโฟลเดอร์ที่คุณเปิดมัน เปิดในโฟลเดอร์ผิด = มันจะแก้ไฟล์ผิดที่ ดังนั้นสร้างโฟลเดอร์ใหม่สำหรับงาน Skill โดยเฉพาะ

1. เปิด Terminal
2. พิมพ์ `cd ~/Desktop` แล้ว Enter (ย้ายไปที่ Desktop)
3. พิมพ์ `mkdir my-skills` แล้ว Enter (สร้างโฟลเดอร์)
4. พิมพ์ `cd my-skills` แล้ว Enter (เข้าโฟลเดอร์)
5. พิมพ์ `claude` แล้ว Enter — Claude Code จะเริ่มทำงาน

> 💡 **วิธีเช็คว่าอยู่โฟลเดอร์ถูกไหม**
>
> พิมพ์ `pwd` ใน terminal จะแสดง path ของโฟลเดอร์ที่คุณอยู่ ถ้าเห็น **/Users/your-name/Desktop/my-skills** ก็ถูกต้อง ถ้าเห็นที่อื่นให้ `cd` ไปให้ถูก

สัญญาณว่า Claude Code เปิดได้

- เห็นข้อความต้อนรับ + logo Claude
- มีช่องให้พิมพ์ prompt อยู่ด้านล่าง
- บอกชื่อโฟลเดอร์ปัจจุบันที่ header

Prompt แรก — ให้ Claude ตรวจเครื่องให้

อย่าเพิ่งสั่ง Claude สร้าง Skill ทันที — ให้มันตรวจเครื่องคุณก่อน เพื่อให้รู้ว่าทุกอย่างพร้อมจริง ๆ และไม่มีอะไรค้างที่จะระเบิดทีหลัง

prompt · ตรวจเครื่องครั้งแรก

```
> ฉันกำลังจะสร้าง Claude Skill แต่ฉัน ไม่รู้โค้ด
ช่วยตรวจเครื่องนี้ให้หน่อยว่า Claude Code พร้อมใช้งานไหม

กติกา:
- อธิบายเป็นภาษาไทยสั้น ๆ
- อย่ารันคำสั่งอันตราย
- ถ้าต้องติดตั้งอะไร ให้ใช้ official docs/source เท่านั้น
- ถ้าไม่แน่ใจ ให้ถามฉันก่อน

หลังตรวจเสร็จ บอกฉันว่า:
1. เครื่องพร้อมไหม
2. ต้องติดตั้งอะไรเพิ่มหรือเปล่า
3. ถ้าต้องติดตั้ง ขออนุมัติฉันก่อน
```

Claude จะตอบกลับมาประมาณนี้

“เครื่องของคุณพร้อมแล้ว — มี Claude Code เวอร์ชัน 2.x โฟลเดอร์ปัจจุบัน ว่างพร้อมสร้างโปรเจกต์ใหม่ ไม่ต้องติดตั้งอะไรเพิ่ม” หรือถ้ายังขาด Claude จะบอกว่า **“ขาด git สำหรับเก็บประวัติ Skill — ขออนุมัติติดตั้งจาก official source ก่อนรันคำสั่ง”**

สรุปบทนี้

- ต้องมี **3 อย่าง**: Claude account, Terminal, Folder
- ติดตั้งจาก **claude.com/code** เท่านั้น — ห้ามใช้ลิงก์อื่น
- เปิด Claude Code ในโฟลเดอร์ที่ **สร้างใหม่** สำหรับงาน Skill
- ทุกครั้งที่ติดตั้งอะไรเพิ่ม ให้ Claude **ขออนุมัติก่อน**

KEY TAKEAWAY

กฎทองของบทนี้ — **ไม่เข้าใจอย่ารัน** ทุกคำสั่งต้อง copy/paste จาก official source เท่านั้น และให้ Claude อธิบายว่ามันทำอะไรก่อนรัน ระบบจะปลอดภัยและคุณจะเข้าใจขึ้นเรื่อย ๆ

→ บทต่อไป
บทที่ 02 · รู้จักโครงสร้าง Skill แบบไม่ต้องอ่านโค้ด
พลิกหน้า →

บทที่ สอง · CHAPTER TWO

02

รู้จักโครงสร้าง Skill แบบไม่ต้องอ่านโค้ด

Skill หนึ่งอันคือ **folder หนึ่งอัน** ในนั้นมีไฟล์เล็ก ๆ ไม่กี่ตัวที่บอก Claude ว่าต้องทำอะไรเมื่อ Skill นี้ถูกเรียกใช้

บทนี้พาทัวร์โครงสร้าง Skill ใน 4 หน้า ไม่ลงไปดูโค้ด แค่รู้ว่า **แต่ละไฟล์มีไว้ทำอะไร** ก็พอจะสร้าง Skill ของตัวเองได้

- ✕ ไม่ต้องจำ syntax ของ YAML / Markdown — Claude เขียนให้
- ✕ ไม่ต้องเขียน script เอง — เริ่มจาก Skill ที่ไม่มี script ก่อน
- ✕ ไม่ต้องเข้าใจ Python — บทนี้ไม่มีโค้ดสักบรรทัด

Skill หนึ่งอัน = folder หนึ่งอัน

เปิด folder ของ Skill ไหนก็ตาม คุณจะเห็นโครงสร้างคล้ายกันหมด — ไฟล์หลักคือ **SKILL.md** และมีโฟลเดอร์ย่อยอีก 2-3 อัน สำหรับเก็บ script / reference / asset

โครงสร้างทั่วไปของ Skill folder

```
brand-voice-rewriter/
├── SKILL.md          # ไฟล์หลัก — บอกว่า Skill นี้คืออะไร
├── references/         # เอกสารอ้างอิง (brand guide ฯลฯ)
│   └── brand-voice.md
├── assets/             # ไฟล์ตัวอย่าง รูป template
│   └── examples.md
└── scripts/            # [optional] Python script ถ้าต้องใช้
    └── helper.py
```

Skill ที่ไม่มี script ก็ใช้งานได้

Skill หลายตัวมีแค่ **SKILL.md** ไฟล์เดียวกับ **references/** ไม่กี่ไฟล์ ก็ทำงานได้ดี เริ่มจากแบบนี้ก่อนเสมอ ค่อยเพิ่ม script ทีหลังเมื่อจำเป็น

SKILL.md — หัวใจของ Skill

ไฟล์เดียวที่ทุก Skill ต้องมี — มันบอก Claude ว่า Skill นี้ **คืออะไร**, **ใช้เมื่อไหร่**, และ **ต้องทำตามขั้นไหน** ส่วนใหญ่จะมีโครงประมาณนี้:

ตัวอย่าง SKILL.md (สั้น)

```
---
name: brand-voice-rewriter
description: ใช้ rewrite ข้อความให้ตรง brand voice ของผู้ใช้
             เรียกใช้เมื่อผู้ใช้ส่งข้อความและขอให้ปรับ tone
---

# Brand Voice Rewriter

## เมื่อไหร่ที่ Skill นี้จะทำงาน
- ผู้ใช้ส่งข้อความและขอให้ "ปรับ tone" / "rewrite"
- ผู้ใช้ขอแคปชั่นโพสต์ในแบบของแบรนด์

## ขั้นตอน
1. อ่าน references/brand-voice.md ก่อนเสมอ
2. rewrite ข้อความ
3. อธิบายว่าปรับอะไรและทำไม
4. เสนอ headline 3 แบบ
```

description คือสิ่งที่สำคัญที่สุด

description คือสิ่งที่ทำให้ Claude รู้ว่า **เมื่อไหร่ควรเรียก Skill นี้** ถ้าเขียนไม่ชัด Claude จะไม่หยิบมาใช้ ถ้าเขียนกว้างเกิน Claude จะเรียกผิดเวลา รายละเอียดเรื่องนี้อยู่ในบทที่ 11

references / scripts / assets ต่างกันยังไง

`references/`	เอกสารที่ Claude อ่านได้ — brand guide, glossary, sample output, do/don't list — เพิ่มได้เรื่อย ๆ
`assets/`	ไฟล์ที่ Skill ใช้ — รูปภาพ, template, sample input — เป็นของจริงที่ Skill อ้างถึง
`scripts/`	Python / shell script ที่ Skill ต้องการเรียกใช้ — สำหรับงานที่ต้องประมวลผล เช่น แตก YouTube ออกเป็นเฟรม

เลือกใส่อะไรในไฟล์ไหน

อ่านอย่างเดียว

ลงใน references/ — Claude จะ load เป็น context

ใช้แสดง

ลงใน assets/ — เป็น file ที่ Skill ส่งกลับให้ผู้ใช้

ต้องประมวลผล

ลงใน scripts/ — Claude จะ run script ผ่าน Bash tool

⚠ **อย่าใส่ secret ใน Skill folder**

ห้ามใส่ API key, password, session token ใน SKILL.md / references / scripts เพราะถ้าคุณ **แชร์ Skill** ให้คนอื่น เขาจะเห็น secret ด้วย เรื่องนี้สำคัญมาก จะพูดอีกครั้งในบทที่ 12

Skill ที่ไม่มี script vs Skill ที่มี script

01

ไม่มี script

เริ่มที่นี่เสมอ · Claude ใช้แค่ความสามารถภาษาธรรมชาติ + อ่าน references · เร็ว ปลอดภัย

02

มี script

เมื่อต้องประมวลผลไฟล์จริง (เช่น แตก video, parse CSV) · ต้องระวัง dependency · ต้องทดสอบมากขึ้น

🎯 **Skill 4 ตัวอย่างใน Part II ของหนังสือนี้**

- **Brand Voice Rewriter** — ไม่มี script
- **Facebook Launch Pack** — ไม่มี script
- **Meeting-to-Action Pack** — ไม่มี script
- **YouTube Content Extractor** — มี script (ใช้ yt-dlp, ffmpeg)

เริ่มจาก 3 ตัวแรกก่อน แล้วค่อยลองตัวที่ 4

ลองให้ Claude อธิบายโครงสร้าง Skill อีกที

ก่อนปิดบทนี้ ลอง prompt นี้กับ Claude ในเครื่องคุณ — จะได้ได้เห็นว่า Claude เข้าใจตรงกับที่หนังสือนี้สอนหรือเปล่า ถ้าตอบไม่ตรง = Claude เปิดดู source ไม่ได้ ต้องเช็คเชื่อมต่ออินเทอร์เน็ต

prompt · ขอให้ Claude อธิบายโครงสร้าง Skill

```
> ช่วยอธิบายโครงสร้าง Claude Skill จาก official spec
ให้ฉันเข้าใจแบบคนไม่รู้โค้ด

ขอเฉพาะส่วนที่ ต้องรู้จริง ในการสร้าง Skill แรก
ไม่ต้องอธิบายเชิงเทคนิคเกินจำเป็น

โปรดอ่าน:
- https://github.com/anthropics/skills/blob/main/skills/skill-
creator/SKILL.md
- https://claude.com/plugins/skill-creator

หลังอ่านเสร็จ สรุปให้ฟังในไม่เกิน 8 บรรทัด
```

KEY TAKEAWAY

Skill = **folder** ที่มี **SKILL.md** เป็นหัวใจ ส่วนที่เหลือ — references, assets, scripts — ใส่เพิ่มเมื่อจำเป็น เริ่มจาก Skill ที่มีแค่ **SKILL.md** ก่อนเสมอ

→ บทต่อไป
บทที่ 03 · ใช้ Skill Creator แบบ Prompt-first
พลิกหน้า →

บทที่ สาม · CHAPTER THREE

03

ใช้ Skill Creator แบบ Prompt-first

Anthropic มี Skill อย่างเป็นทางการชื่อว่า **Skill Creator** ที่ออกแบบมาเพื่อสร้าง Skill ใหม่ — มันรู้ structure, best practice, และจุดที่มือใหม่พลาดบ่อย

บทนี้สอนให้คุณ **ใช้ Skill Creator** ช่วยสร้าง Skill แรก โดยไม่ต้องเปิดไฟล์ใด ๆ ด้วยตัวเอง — คุยกับ Claude เป็นหลัก

- ✕ ไม่ต้องเขียน SKILL.md เอง — Skill Creator เขียนให้
- ✕ ไม่ต้อง clone repo — Claude Code ดึงให้ผ่าน Skills ในตัว
- ✕ ไม่ต้องตั้ง YAML / Markdown frontmatter — Skill Creator จัดให้

Skill Creator คืออะไร

Skill Creator คือ **Skill ที่ใช้สร้าง Skill** — อยู่ใน official repo ของ Anthropic ชื่อ `anthropics/skills` ภายใต้โฟลเดอร์ `skills/skill-creator/`

01

ทำหน้าที่อะไร

ถาม requirement → propose โครงสร้าง → เขียน SKILL.md → review → ปรับ → ทดสอบ

02

ทำไมต้องใช้

เพราะมัน enforce best practice ของ Skill spec อัตโนมัติ · ไม่ต้องจำเองว่าต้องมี field อะไร

💡 **Skill Creator vs เริ่มเขียนเอง**

มือใหม่ที่เริ่มเขียน SKILL.md เองมักลืม **description** ที่ทำให้ trigger ทำงาน หรือเขียน **instructions** กว้างเกิน Skill Creator ถามรายละเอียดให้ครบก่อนเขียน — ผลลัพธ์ใช้งานได้จริงตั้งแต่ครั้งแรก

ทำไมไม่ควรเริ่มจากเขียนไฟล์เอง

การ copy SKILL.md จากหนังสือมาเริ่มเอง — ดูเหมือนเร็วกว่า แต่จริง ๆ ช้ากว่า เพราะคุณจะไม่ได้ **คิดให้ครบ** ก่อนเขียน Skill Creator ถามคุณเหมือนนักออกแบบ — ผลคือ Skill ที่ **ตรงกับงานคุณ** ไม่ใช่ template ที่ลอกมา

วิธีให้ Claude Code อ่าน Skill Creator

1. เปิด Claude Code ในโฟลเดอร์ project ของคุณ
2. ส่ง prompt ที่หน้าถัดไป — มัน **ชี้ official source** และบอก context ให้ Claude
3. Claude จะดาวน์โหลด / อ่าน Skill Creator จาก official repo
4. Claude สรุปกลับมาเป็นภาษาไทยว่าเข้าใจอะไร แล้วถามคำถามต่อ
5. คุณตอบคำถาม → Skill Creator propose plan → คุณอนุมัติ → เขียน

> ⚠️ **อย่าให้ Claude สร้างไฟล์ก่อนคุณอนุมัติ**
>
> ใน prompt ครั้งแรก ต้องเขียนชัด ๆ ว่า **"ห้ามสร้างไฟล์จนกว่าจะสรุปแผนให้ฉันอนุมัติ"** ไม่งั้น Claude อาจ jump ไปเขียนเลย และคุณจะได้ Skill ที่ไม่ตรงกับที่คิด

Prompt หลัก — เรียก Skill Creator มาช่วย

นี่คือ prompt ที่ใช้จริงทุกครั้งที่คุณอยากเริ่ม Skill ใหม่ copy ทั้งกล่องนี้ไปวางใน Claude Code แล้วเติม **“Skill ที่อยากสร้าง”** ตอนท้าย

prompt · เปิด Skill Creator + สร้าง Skill ใหม่

```
> ฉันต้องการสร้าง Claude Skill ใหม่ โดยใช้ official Skill Creator
เป็นแนวทาง

โปรดอ่าน sources เหล่านี้ก่อน:
- https://claude.com/plugins/skill-creator
- https://github.com/anthropics/skills/blob/main/skills/skill-
creator/SKILL.md

กติกา:
- ฉันไม่รู้โค้ด
- อธิบายเป็นภาษาไทย
- ห้ามสร้างไฟล์จนกว่าจะสรุปแผนให้ฉันอนุมัติ
- ใช้ official docs/repo เท่านั้น
- ถ้าข้อมูลไม่ครบ ให้ถามฉันทีละ 3 คำถาม

Skill ที่อยากสร้างคือ:
[ใส่งานที่อยากให้ Skill ทำ เช่น "rewrite ข้อความให้ตรง brand voice ของฉัน"]
```

Claude จะทำอะไรหลังได้ prompt

1. เปิดอ่าน Skill Creator จาก official repo (ใช้เวลา 10-30 วินาที)
2. สรุปกลับมาเป็นภาษาไทยว่า **เข้าใจอะไร**
3. ถามคำถาม **ทีละ 3 ข้อ** เพื่อให้ Skill brief แน่นพอ
4. เสนอ **โครงสร้าง folder + SKILL.md draft**
5. รอคุณอนุมัติ → จึงสร้างไฟล์
6. หลังสร้างเสร็จ ทดสอบให้ดูตัวอย่าง 1 case

🎯 **ตัวอย่างคำถามที่ Claude จะถาม**

- Skill นี้ trigger เมื่อไหร่? (เช่น เมื่อผู้ใช้ส่งข้อความและขอ rewrite)
- Input ต้องการอะไรบ้าง? (ข้อความ + reference?)
- Output format ที่ต้องการคืออะไร? (มี section อะไรบ้าง?)
- ถ้าข้อมูลไม่ครบ Skill ควรทำยังไง? (ถามกลับ? ใช้ default?)
- มีตัวอย่าง output ที่ดีที่อยากให้ Claude อ้างอิงไหม?

สรุปบทนี้

- Skill Creator = **Skill อย่างเป็นทางการ** ที่ใช้สร้าง Skill อื่น
- อย่าเขียน SKILL.md เอง — ให้ Skill Creator **ถามและเขียน** ให้
- ใน prompt ครั้งแรก ต้องบอก **ห้ามสร้างไฟล์จนกว่าจะอนุมัติ**
- ตอบคำถาม Claude ให้ละเอียด → ได้ Skill ที่ใช้งานได้ตั้งแต่ครั้งแรก

KEY TAKEAWAY

Skill Creator คือ **นักออกแบบ Skill ส่วนตัว** ของคุณ — หน้าที่คุณคือตอบคำถามให้ชัด หน้าที่ Claude คือเขียนไฟล์ให้ถูก spec คุณไม่ต้องเปิดไฟล์ใด ๆ เองตลอดบทนี้

→ บทต่อไป
บทที่ 04 · สูตรคิด Skill ที่ดี: งานแคบ ชัด ใช้ซ้ำได้
พลิกหน้า →

บทที่ สี่ · CHAPTER FOUR

04

สูตรคิด Skill ที่ดี: งานแคบ ชัด และ ใช้ซ้ำได้

Skill ที่ดีไม่ได้เกิดจากการ **“สั่ง Claude ให้ทำอะไรก็ได้”** แต่เกิดจาก **การออกแบบ workflow ก่อน** แล้วค่อยให้ Claude สร้าง

บทนี้สอนสูตร **Skill Brief 7 ช่อง** ที่เปลี่ยนไอเดียกว้าง ๆ ในหัวให้กลายเป็นโจทย์ที่ชัดพอจะสร้างได้ สูตรนี้ใช้กับทุก Skill ใน Part II ของหนังสือเล่มนี้

- ✕ ไม่ใช่สูตรเขียน prompt — เป็นสูตรออกแบบ workflow
- ✕ ไม่ใช่ template ตายตัว — ปรับให้เข้ากับงานคุณได้
- ✕ ไม่ใช่การจำกัด Claude — เป็นการบอก Claude ว่า “ดี” คืออะไร

Skill ที่ดีต้องมี trigger ชัด

Trigger คือ **สัญญาณ** ที่ทำให้ Claude รู้ว่า — “อ๋อ ตอนนี้ผู้ใช้ขอสิ่งที่ Skill ตัวนี้ทำ ฉันควรเรียกมัน” Trigger ที่ดี = Skill ทำงานถูกเวลา

✕

Trigger คลุมเครือ

“ช่วยทำ marketing” — ไม่รู้ว่าผู้ใช้ขออะไรกันแน่ Claude ลังเลและไม่หยิบ Skill มาใช้

✓

Trigger ชัด

“ผู้ใช้ส่ง YouTube URL และขอ content pack” — มี keyword (URL + content pack) ที่แม่นยำ

💡 **วิธีคิด Trigger 3 ขั้น**

1. ผู้ใช้พิมพ์อะไรเข้ามา? (คำเฉพาะ / URL / file type)
2. ผู้ใช้คาดหวัง output แบบไหน? (post / list / file)
3. มีสัญญาณอะไรที่บอกว่า **“อันนี้ Skill ตัวนี้แน่ ไม่ใช่ตัวอื่น”**?

อย่าทำ Skill กว้างเกิน

อาการที่พบบ่อยของผู้เริ่มต้น — อยากให้ Skill **“ทำได้ทุกอย่าง”** เพราะคิดว่ายิ่งกว้างยิ่งคุ้ม ผลคือกลายเป็น Skill ที่ Claude **ไม่หยิบมาใช้เลย** เพราะไม่รู้ตอนไหนควรใช้

กว้างเกิน: “ดูแล social ให้”

→ ปรับเป็น: “แปลง long-form video เป็น Facebook post pack 3 แบบ”

กว้างเกิน: “ช่วยจัดการ email”

→ ปรับเป็น: “จัด priority email ลูกค้าตอนเช้า + draft reply 3 ฉบับแรก”

กว้างเกิน: “ทำ content”

→ ปรับเป็น: “เปลี่ยน research report เป็น carousel 7-10 slide พร้อม source notes”

> ⚠️ **กฎ 1 ประโยค**
>
> ถ้าคุณอธิบาย Skill นี้ไม่ได้ใน **1 ประโยคที่ชัด** — Skill ยังกว้างเกิน ปรับโจทย์ก่อนลงมือ

Skill Brief 7 ช่อง — สูตรหลักของหนังสือเล่มนี้

Skill ทุกตัวใน Part II ของหนังสือเล่มนี้ออกแบบผ่าน 7 ช่องนี้ ก่อนจะลงมือสร้าง — กรอกให้ครบทุกช่อง ใช้เวลา 10-15 นาที แต่ประหยัดเวลา debug ภายหลังได้หลายชั่วโมง

01
ทำอะไร
1 ประโยค บอกหน้าที่ Skill นี้แบบเฉพาะเจาะจง

02
ใช้เมื่อไหร่
trigger ที่ Claude จะรู้ว่าเรียก Skill นี้

03
Input
ผู้ใช้ต้องส่งอะไรเข้ามาบ้าง (text / URL / file)

04
Output
ต้องได้อะไรกลับ + รูปแบบที่แน่นอน (sections + format)

05
Workflow
ขั้นตอนทำงานของ Skill — ทำอะไรก่อน / อะไรหลัง

06
ห้ามทำ
ข้อจำกัด / สิ่งที่ Claude ไม่ควรทำ

07 ตัวอย่างผลลัพธ์ที่ดี

ตัวอย่าง output 1-2 ชุดที่คุณคิดว่า **“แบบนี้แหละ”** ใส่ใน `references/examples.md` ให้ Claude อ้างอิงได้ตอนสร้างของจริง — เป็นช่องที่สำคัญที่สุด เพราะมัน **นิยามคำว่า “ดี”** ให้ Skill

Prompt — ให้ Claude ช่วยกรอก Skill Brief

ถ้าคุณยังคิดไม่ออกว่าจะกรอก 7 ช่องยังไง ใช้ prompt นี้กับ Claude บอกแค่ **ไอเดียกว้าง ๆ** Claude จะถามต่อทีละช่องและช่วยปรับให้ชัด

prompt · เปลี่ยนไอเดียเป็น Skill Brief 7 ช่อง

```
> ช่วยเปลี่ยนไอเดียนี้ให้เป็น Skill Brief 7 ช่อง
ที่ชัดพอสำหรับสร้าง Claude Skill

ไอเดีย: [ใส่ไอเดียกว้าง ๆ ที่อยากทำ]

ขอ output เป็น 7 ส่วน:
1. Skill นี้ทำอะไร (1 ประโยค)
2. ใช้เมื่อไหร่ (trigger)
3. input ที่ต้องการ
4. output ที่ต้องได้ (รวม format)
5. workflow ทีละขั้น
6. ข้อห้าม / ข้อควรระวัง
7. ตัวอย่าง prompt เวลาเรียกใช้ Skill

ถ้าไอเดียยังไม่ชัด ถามฉันก่อน ทีละ 2-3 คำถาม
ห้ามเดาเอง
```

สรุปบทนี้

- Skill ที่ดี = **trigger ชัด + งานแคบ + output แน่นอน**
- กฎ 1 ประโยค — ถ้าอธิบายไม่ได้ในประโยคเดียว ปรับโจทย์ก่อน
- ใช้ **Skill Brief 7 ช่อง** ก่อนลงมือทุกครั้ง
- ถ้าคิดไม่ออก ให้ Claude ช่วยถาม + กรอกให้ — ใช้ prompt ในหน้า 5

KEY TAKEAWAY

การใช้เวลา 15 นาทีกรอก Skill Brief 7 ช่อง **ประหยัดเวลา debug** Skill ทีหลังได้หลายชั่วโมง — เพราะ Claude มี **เกณฑ์ที่ชัด** ในการตัดสินใจตลอดทาง ไม่ใช่เดาเอาเอง

→ บทต่อไป · เริ่ม PART II
บทที่ 05 · Brand Voice Rewriter — Skill แรกแบบไม่มี script
ลงมือ →

ส่วนที่ 2

II

6 Skill ตัวอย่างใช้ได้จริง

แต่ละบทสร้าง Skill ที่จับต้องได้ ใช้กับงานครีเอเตอร์ การตลาด freelancer และเจ้าของธุรกิจขนาดเล็ก

บทที่ ห้า · CHAPTER FIVE

05

Brand Voice Rewriter — Skill แรกแบบไม่มี script

นี่คือ Skill ตัวจริงตัวแรกที่คุณจะสร้าง — เรียบง่าย ไม่มี script แต่ **ใช้งานได้ทุกวัน** สำหรับคนที่โพสต์ขายของ ทำเพจ หรือส่งงาน copywriting ให้ลูกค้า

เริ่มต้นจาก Skill ที่ **ไม่มี script** เป็นการเริ่มที่ปลอดภัยที่สุด — ไม่ต้องติดตั้งอะไรเพิ่ม ทดสอบเร็ว และเห็นผลทันที

- ✕ ไม่ต้องเขียน script Python — Skill นี้ใช้แค่ SKILL.md + references
- ✕ ไม่ต้องมี brand guide แบบเป็นทางการ — ตัวอย่าง 5-10 ประโยคก็พอ
- ✕ ไม่ต้องเก่งภาษาอังกฤษ — Skill ทำงานภาษาไทยเต็มที่

Brand Voice Rewriter — Brief 7 ช่อง

ก่อนสร้าง Skill ใด ๆ ในเล่มนี้ เราจะกรอก Skill Brief 7 ช่องจากบทที่ 4 ก่อน เสมอ Brief นี้คือสิ่งที่คุณจะส่งให้ Skill Creator (ที่ตั้งไว้ในบทที่ 3) เป็น input

01. ทำอะไร	Rewrite ข้อความให้ตรง brand voice ของผู้ใช้ — ปรับ tone, ตัดน้ำ, ปิดด้วย CTA
02. ใช้เมื่อไหร่	ผู้ใช้ส่งข้อความและขอ "ปรับ tone", "rewrite", "ทำให้เป็นแบบของเรา"
03. Input	ข้อความที่ต้องการ rewrite + ระบุ platform (Facebook / IG / LINE OA)
04. Output	เวอร์ชัน rewrite + เหตุผลที่ปรับ + headline 3 แบบ
05. Workflow	(1) อ่าน references/brand-voice.md (2) วิเคราะห์ tone เดิม (3) rewrite (4) explain (5) suggest headlines
06. ห้ามทำ	ห้ามใช้คำเวอร์ · ห้ามใช้ emoji เกิน 2 ตัว · ห้ามแต่ง claim ที่ผู้ใช้ไม่ได้ระบุ
07. ตัวอย่าง	ตัวอย่าง before/after 3 ชุด เก็บใน references/examples.md

เตรียม brand voice examples ก่อนสร้าง Skill

Skill นี้ **คุณภาพขึ้นกับ examples** — มากกว่า prompt engineering ใช้เวลา 30 นาทีเก็บตัวอย่าง 5-10 ประโยคที่เป็น **brand voice ของคุณจริง ๆ** ก่อนเริ่มสร้าง Skill

01

ดึงจากโพสต์เก่า

เปิดเพจ/IG ของคุณ copy 5 โพสต์ที่คุณคิดว่าตรง brand voice ที่สุด — เป็น "ตัวอย่างที่ดี"

02

เขียนสิ่งที่ห้าม

ใส่ 3-5 ตัวอย่างของ tone ที่ห้ามใช้ — เป็น "ตัวอย่างที่ไม่ใช่" สำคัญพอ ๆ กัน

Template brand-voice.md

เก็บไว้ใน `references/brand-voice.md` โครงประมาณนี้:

- **Brand persona** — เราคือใคร พูดยังไง (3-4 บรรทัด)
- **Do** — คำ/วิธีพูดที่ใช้ + ตัวอย่าง 5 ชุด
- **Don't** — คำที่ห้ามใช้ + ตัวอย่างที่ไม่ใช่ 3 ชุด
- **Platform tone** — Facebook vs IG vs LINE OA ต่างกันยังไง

Prompt — สร้าง Skill brand-voice-rewriter

เมื่อกรอก Brief และเตรียม examples แล้ว ส่ง prompt นี้ใน Claude Code ที่อยู่ในโฟลเดอร์ `my-skills/` ของคุณ Skill Creator จะอ่าน Brief ของคุณและเขียน Skill ให้

prompt · สร้าง Brand Voice Rewriter

```
> ช่วยสร้าง Claude Skill ชื่อ brand-voice-rewriter

Skill นี้ใช้สำหรับ rewrite ข้อความภาษาไทย
ให้ตรง brand voice ของฉัน

Brand voice:
- เป็นกันเอง ฉลาดแต่ไม่อวดเก่ง
- ขายได้แต่ไม่ hard sell
- ประโยคสั้น กระชับ
- เหมาะกับ Facebook post

Output ต้องมี:
1. เวอร์ชัน rewrite
2. เหตุผลว่าปรับอะไร
3. ตัวเลือก headline 3 แบบ

โปรดสร้าง Skill แบบไม่มี script ก่อน
ใช้ official Skill Creator
ห้ามสร้างไฟล์จนกว่าจะสรุปแผนให้ฉันอนุมัติ
```

ทดสอบ Skill ด้วยข้อความจริง

หลัง Skill ถูกสร้างเสร็จ อย่าเพิ่ง declare ว่า “เสร็จแล้ว” — ทดสอบ 3 case นี้ก่อนใช้งานจริง

1. **Case ง่าย** — copy โพสต์เก่าของคุณมา rewrite ดูว่ายังตรง tone ไหม
2. **Case ใหม่** — เขียนข้อความแย่ ๆ ดูว่า Skill ปรับขึ้นจริงไหม
3. **Case กวน** — ส่งข้อความที่ไม่ระบุ platform ดูว่า Skill ถามกลับไหม

🎯 **ถ้า Skill ไม่ถามกลับใน case 3 → ปรับ description**

Skill ที่ดีต้อง **ถามกลับเมื่อข้อมูลไม่ครบ** ไม่ใช่เดาเอา ถ้าคุณส่งข้อความที่ไม่ระบุ Facebook/IG/LINE แล้วมัน rewrite ทันที — ให้ Claude แก้ **description** + **workflow** ใน SKILL.md ให้บังคับให้ถามก่อน

Prompt สำหรับให้ Claude ทดสอบและปรับ Skill

prompt · ทดสอบ + grade + ปรับ

```
> ช่วยทดสอบ Skill brand-voice-rewriter ด้วย 3 test case
ที่หน้านี้บอก

ให้ประเมิน output แต่ละ case แล้วเสนอว่าควรแก้
description หรือ workflow ใน SKILL.md ตรงไหน
ก่อนแก้ ขออนุมัติฉันก่อน
```

สรุปบทนี้

- Brand Voice Rewriter = **Skill แรก** ที่ควรลองสร้าง — ไม่มี script ไม่ต้อง dependency
- คุณภาพ Skill นี้ขึ้นกับ **examples** มากกว่า prompt — ลงทุน 30 นาทีเก็บตัวอย่าง
- หลังสร้างต้องทดสอบ 3 case (ง่าย / ใหม่ / กวน) ก่อนใช้จริง
- Skill ที่ดี **ถามกลับเมื่อข้อมูลไม่ครบ** — ถ้าไม่ถาม ให้ปรับ description

KEY TAKEAWAY

Skill ที่ไม่มี script เริ่มเร็วและเห็นผลทันที — ไม่ต้องกังวลเรื่อง dependency, scripts, หรือ tool installation เริ่มจากที่นี่ ค่อยขยายไปสู่ Skill ที่ใหญ่กว่าในบทถัดไป

→ บทต่อไป
บทที่ 06 · YouTube Content Extractor — Skill ที่มี script
พลิกหน้า →

บทที่ หก · CHAPTER SIX

06

YouTube Content Extractor

ส่ง URL วิดีโอเดียว ได้ **caption + summary + post pack** พร้อมใช้ — Skill นี้คือตัวอย่างแรกที่มี script เพราะต้อง ดาวน์โหลด/แปลงไฟล์จริง

แต่ **คุณยังไม่ต้องเขียน script เอง** — บอก Claude ว่า ต้องการ tool อะไร แล้วให้มัน **research + install** จาก official source ให้

- ✕ ไม่ละเมิดลิขสิทธิ์ — ใช้กับวิดีโอที่คุณมีสิทธิ์วิเคราะห์เท่านั้น
- ✕ ไม่ต้องเขียน script เอง — Claude จัดให้ตาม official docs
- ✕ ไม่ใช้กับวิดีโอที่ลงทะเบียน private — yt-dlp ไม่รองรับ และมัน infringe ToS

YouTube Content Extractor — Brief 7 ช่อง

01. ทำอะไร	รับ YouTube URL → สร้าง content pack: caption + summary + key moments + thumbnails + Facebook posts + Reels caption
02. ใช้เมื่อไหร่	ผู้ใช้ส่ง YouTube URL และขอ "content pack" / "caption pack" / "แตกโพสต์จากวิดีโอ"
03. Input	YouTube URL (ที่ผู้ใช้มีสิทธิ์ใช้) + optional: target audience, post tone
04. Output	transcript.txt + summary.md + key-moments.md + thumbnails/*.jpg + fb-posts.md + reels-caption.md
05. Workflow	(1) ตรวจสิทธิ์ใช้งาน (2) download audio + transcribe (3) extract frames (4) gen summary (5) gen posts
06. ห้ามทำ	ห้าม download private video · ห้ามแต่ง quote ที่ไม่มีในวิดีโอ · ห้ามใส่ thumbnail ที่มีลิขสิทธิ์ผู้อื่น
07. ตัวอย่าง	Output folder ตัวอย่าง 1 ชุดจากวิดีโอ public domain เก็บใน references/example-output/

Tools ที่ Skill นี้ต้องใช้

Skill นี้ต่างจาก ch05 — มันต้องใช้ **2 tool ภายนอก** ในการดาวน์โหลดและแปลงไฟล์ ทั้งคู่เป็น open-source

yt-dlp

Download video/audio จาก YouTube · ใช้ผ่าน command line · มี Python module ในตัว · official: github.com/yt-dlp/yt-dlp

ffmpeg

แปลง audio/video formats + extract frames · มาตรฐานอุตสาหกรรม · official: ffmpeg.org

ห้ามให้ Claude ติดตั้งโดยไม่อนุมัติ

ทั้งคู่ติดตั้งง่าย แต่ **ห้าม**ให้ Claude รัน install command โดยที่คุณยังไม่ได้อนุมัติ — ในหน้าถัดไปมี prompt ที่บอก Claude ให้ **"ขออนุมัติก่อนทุก install"** อย่างชัดเจน

วิธีตัดสินใจว่าควรอนุมัติให้ติดตั้งหรือไม่

1. Claude แสดง URL ของ tool — เปิดตรวจว่าเป็น **official** ไหม
2. ดูคำสั่งติดตั้ง — ปกติจะใช้ `brew` / `apt` / `pip`
3. ถามตัวเองว่า — "tool นี้ **ทำอะไรบ้าง** นอกจากที่ Skill ต้องการ?"
4. ถ้าทุกข้อปลอดภัย — อนุมัติ

Prompt — สร้าง youtube-content-extractor

prompt · สร้าง Skill + ให้ Claude research tool

```
> ช่วยสร้าง Claude Skill ชื่อ youtube-content-extractor

หน้าที่:
เมื่อฉันส่ง YouTube URL ให้ Skill นี้ช่วยสร้าง content pack
สำหรับนำไปทำโพสต์และศึกษาเนื้อหา

Output ที่ต้องการ:
- transcript / full caption (.txt)
- summary ภาษาไทย
- key moments + timestamp
- thumbnails จากช่วงสำคัญ (3-5 ภาพ)
- Facebook post 3 แบบ
- short-form caption 5 แบบ

กติกา:
- ใช้กับวิดีโอที่ฉันมีสิทธิ์ใช้งาน เท่านั้น
- ถ้าต้องติดตั้ง tool เพิ่ม ให้ใช้ official source เท่านั้น
- ก่อนติดตั้งอะไร ขออนุมัติฉันก่อน
- ฉันไม่รู้โค้ด — ทำเป็นขั้นตอนและอธิบายสั้น ๆ
- ใช้ official Skill Creator เป็นแนวทาง
```

Workflow ของ Skill นี้ทำงานยังไง

1. ผู้ใช้ส่ง URL — Skill **ตรวจสิทธิ์ใช้งาน** ก่อนเสมอ
2. Claude เรียก script ที่ใช้ `yt-dlp` ดาวน์โหลด audio + metadata
3. ส่ง audio ให้ Claude transcribe เป็น text
4. `ffmpeg` extract 5-10 frame สำคัญเป็น thumbnails
5. Claude อ่าน transcript → สร้าง summary, key moments, posts
6. เก็บทุกอย่างใน folder `output/[video-id]/`

ตัวอย่าง output folder

```
output/abc123/
├── transcript.txt          (12,400 คำ)
├── summary.md              (สรุปไทย 6 ย่อหน้า)
├── key-moments.md          (8 จุดสำคัญ + timestamp)
├── thumbnails/
│   ├── 00-intro.jpg
│   ├── 02-main-point-a.jpg
│   └── 05-conclusion.jpg
├── fb-posts.md             (3 แบบ พร้อม headline)
└── reels-caption.md        (5 แบบ พร้อม hashtag)
```

สรุปบทนี้

- Skill นี้คือ **ตัวอย่างแรกที่มี script** — แต่คุณยังไม่ต้องเขียน script เอง
- ใช้ tool ภายนอก: **yt-dlp + ffmpeg** — official open-source ทั้งคู่
- ทุก install command ต้อง **ขออนุมัติคุณก่อนรัน**
- ห้ามใช้กับวิดีโอที่คุณไม่มีสิทธิ์ — กฎข้อนี้เคารพ ทั้งทางกฎหมายและจริยธรรม

KEY TAKEAWAY

Skill ที่มี script เปิดโอกาส **มากกว่า** Skill ไม่มี script — แต่ก็มี responsibility มากกว่า ใช้กฎ **"official source + ขออนุมัติก่อน install"** ตลอดทาง คุณจะเริ่มเชื่อใจ Claude มากขึ้นเรื่อย ๆ

→ บทต่อไป
บทที่ 07 · Facebook Launch Pack Generator
พลิกหน้า →

บทที่ เจ็ด · CHAPTER SEVEN

07

Facebook Launch Pack Generator

Skill สำหรับ **creator / founder / agency** ที่ต้องเปิดตัวสินค้าหรือบริการบ่อย — รับข้อมูลครั้งเดียว ได้ชุดโพสต์เปิดตัวพร้อมใช้ทันที

กลับมาที่ Skill แบบ **ไม่มี script** — พลังของ Skill นี้อยู่ที่ **โครง output ที่แน่นอน** และ **tone ที่ตรงกลุ่ม**

- ✕ ไม่ทำ headline หลอกลวง / clickbait เกินจริง
- ✕ ไม่แต่ง claim ที่ผู้ใช้ไม่ระบุ — ห้าม “รับประกันคืนเงิน 100%” ถ้าไม่ได้บอก
- ✕ ไม่ละเมิดสิทธิ์ความเป็นส่วนตัวของลูกค้าตัวอย่าง

Facebook Launch Pack — Brief 7 ช่อง

01. ทำอะไร	สร้างชุดโพสต์เปิดตัวสินค้า/บริการ พร้อม founder story, FAQ, objection handling, CTA
02. ใช้เมื่อไหร่	ผู้ใช้บอก "จะ launch สินค้า X" หรือส่งข้อมูลสินค้าและขอ "ชุดโพสต์เปิดตัว"
03. Input	ชื่อ + คุณสมบัติสินค้า + กลุ่มลูกค้า + ราคา + USP + วันเปิดตัว
04. Output	launch posts (3) + founder story + FAQ + objection handling + caption สั้น (10) + CTA + comment reply template
05. Workflow	(1) ถามข้อมูลที่ขาด (2) วิเคราะห์ USP (3) สร้าง launch posts (4) สร้าง support content (5) สรุปเป็น Markdown
06. ห้ามทำ	ห้าม headline หลอกลวง · ห้าม claim เกินจริง · ห้ามใส่ราคาที่ผู้ใช้ไม่ระบุ · ห้าม emoji เกิน 3 ตัว/โพสต์
07. ตัวอย่าง	ตัวอย่าง launch pack 2 ชุด จาก case study จริง (anonymized)

Output ที่ Skill นี้ต้องส่งกลับ

Skill นี้ **ขายดี** เพราะมัน output เยอะและตรงงาน ลองดูรายการเต็ม — ทุกอย่างนี้เกิดจาก input ก้อนเดียว

📣

Launch Post (3 แบบ)

(1) Hook-driven · (2) Story-driven · (3) Problem-Solution — เลือกใช้ตามวันเปิดตัวและ momentum

👤

Founder Story Post

เล่าเรื่องผู้ก่อตั้ง / ทีม / ที่มาของสินค้า — เน้นความจริงไม่เวอร์

❓

FAQ Post

5-7 คำถามที่ลูกค้าน่าจะถาม + คำตอบสั้น พร้อม source

🛡️

Objection Handling

คำตอบสำหรับข้อกังวลที่พบบ่อย เช่น ราคา ขนาด การใช้งาน การคืนเงิน

✨

Caption สั้น (10 แบบ)

สำหรับ Reels / TikTok / Shorts — มี hook + body + CTA สั้นในแต่ละแบบ

💬

Comment Reply Template

ตัวอย่างตอบ comment 5 แบบ ตาม sentiment ลูกค้า

Prompt — สร้าง facebook-launch-pack-generator

prompt · สร้าง Skill

```
> ช่วยสร้าง Claude Skill ชื่อ facebook-launch-pack-generator

Skill นี้รับข้อมูลสินค้า/บริการ
แล้วสร้างชุดโพสต์เปิดตัวสำหรับ Facebook

Output ต้องมี:
1. launch post 3 สไตล์ (hook / story / problem-solution)
2. founder story post
3. FAQ post (5-7 คำถาม)
4. objection handling post
5. caption สั้น 10 แบบ
6. comment reply template (5 แบบ)
7. CTA หลายแบบตาม channel

กติกา:
- ภาษาไทย กระชับ ไม่เวอร์
- ไม่หลอกลวง ไม่ใช้ claim เกินจริง
- ถ้าข้อมูลขาด ถามก่อน ทีละ 3 ข้อ
- ใช้ official Skill Creator
- ห้ามสร้างไฟล์จนกว่าจะสรุปแผนให้ฉันอนุมัติ
```

ทำไม Skill นี้ขายดี (และทำ skill pack ต่อยอดได้)

ขายของออนไลน์

เปิดตัวสินค้าใหม่ทุกเดือน — ใช้ Skill นี้ลดเวลาทำคอนเทนต์ครึ่งหนึ่ง

Agency

ส่งงาน launch ให้ลูกค้าเร็วขึ้น — ใช้เป็น first draft ก่อนปรับเข้า brand

Founder

ทำ launch โดยไม่ต้องจ้าง copywriter — เหมาะกับ MVP / soft launch

💡 **ต่อยอดเป็น skill pack ขาย/แจกได้**

Skill เดียวขายยาก — แต่ **3-5 Skill รวมกัน** เป็น **"Launch Skill Pack"** มีค่ามาก ตัวอย่าง pack:

- facebook-launch-pack-generator (บทนี้)
- brand-voice-rewriter (บทที่ 5)
- youtube-content-extractor (บทที่ 6)
- email-sequence-builder (ต่อยอดเอง)

วิธีแพ็กและขาย — บทที่ 12

สรุปบทนี้

- Skill นี้ **กลับมาเป็นแบบไม่มี script** — แต่ output เยอะกว่า ch05
- พลังของมันอยู่ที่ **โครง output ที่แน่นอน** ไม่ใช่ความซับซ้อนของ code
- กฎทอง: **ห้ามแต่ง claim ที่ผู้ใช้ไม่ระบุ** — Skill จะตอบจากข้อเท็จจริงเท่านั้น
- ต่อยอดเป็น **Skill Pack** ขาย/แจกได้ (อ่านบทที่ 12)

KEY TAKEAWAY

Skill ที่ขายได้ **ไม่ใช่ Skill ที่เก่งที่สุด** — แต่เป็น Skill ที่ **"ตัด workflow ที่คนทำซ้ำๆ ในชีวิตประจำวัน"** ออกได้ครึ่งหนึ่ง Facebook launch คืองานที่ทุกคนทำซ้ำเดือนละครั้ง นั่นคือเหตุผลที่ Skill นี้ขายดี

→ บทต่อไป
บทที่ 08 · Meeting-to-Action Pack
พลิกหน้า →

บทที่ แปด · CHAPTER EIGHT

08

Meeting to-Action Pack

Skill ที่ **ทุกทีมต้องใช้** — รับ transcript หรือ note จาก meeting แปลงเป็น action plan พร้อมส่งให้ทีม

ไม่มี script แต่ทำงานได้ทั้งภาษาไทยและอังกฤษ — ถ้ามี audio recording ค่อยต่อยอด integrate กับ transcription tool ในอนาคต

- ✕ ไม่ใช่ Skill อัด audio — รับ text เป็นหลัก
- ✕ ไม่ส่ง action items อัตโนมัติเข้า PM tool — ให้ผู้ใช้ตรวจก่อน
- ✕ ไม่เปลี่ยน decision ของทีม — แค่จัดให้เป็นรูปแบบที่ส่งต่อได้

Meeting-to-Action Pack — Brief 7 ช่อง

01. ทำอะไร	แปลง meeting transcript / note → action plan + follow-up email + next agenda
02. ใช้เมื่อไหร่	ผู้ใช้ส่ง meeting note และขอ "สรุป meeting" / "action items" / "follow up"
03. Input	transcript หรือ note (ภาษาไทย/อังกฤษ) + optional: ชื่อผู้ร่วมประชุม / context project
04. Output	summary + decisions + action items (owner + deadline) + risks + follow-up email + next agenda + chat message
05. Workflow	(1) อ่าน + แยก speaker (2) หา decisions (3) extract action items (4) gen email + agenda
06. ห้ามทำ	ห้ามแต่ง action ที่ไม่มีในต้นฉบับ · ห้ามใส่ deadline เอง · ห้ามตัดสินใจแทนทีม
07. ตัวอย่าง	transcript ตัวอย่าง 2 ชุด (anonymized) + sample output ที่ "ดี" + "ไม่ดี"

Output ที่ Skill ต้องส่งกลับ

📋
Executive Summary
3-5 บรรทัด สำหรับผู้บริหารหรือคนที่ไม่ได้เข้าประชุม

✅
Decisions Made
รายการการตัดสินใจที่ทีมตกลงกัน — แยกจาก discussion ที่ยังเปิดอยู่

🎯
Action Items
งาน + owner + deadline (ตรงจากต้นฉบับเท่านั้น)

⚠️
Risks / Blockers
อุปสรรคที่ทีมพูดถึง — ใครต้องจัดการ

✉️
Follow-up Email
draft อีเมล follow-up ที่ส่งให้ทุกคนใน meeting

📅
Next Agenda
agenda ของ meeting ครั้งหน้า (ถ้ามี)

💡 **Chat message version**

ใส่เพิ่ม 1 ส่วน — **“short message สำหรับส่งใน LINE/Slack/Discord”** เป็นเวอร์ชันสั้นกว่า email มี 3-5 bullet ส่งทันทีหลังประชุมจบ เหมาะกับทีมที่ใช้ chat เป็นหลัก ไม่ค่อยเปิด email

Prompt — สร้าง meeting-to-action-pack

prompt · สร้าง Skill

```
> ช่วยสร้าง Claude Skill ชื่อ meeting-to-action-pack

Skill นี้ใช้แปลง meeting notes หรือ transcript
ให้เป็น action plan ภาษาไทย

Output ต้องมี:
- executive summary
- decisions made
- action items พร้อม owner / deadline
- risks / blockers
- follow-up email
- next meeting agenda
- short message สำหรับส่งใน chat (LINE/Slack)

กติกา:
- ถ้าข้อมูลไม่พอ ถามกลับก่อนสรุป
- ห้ามแต่ง action ที่ไม่มีในต้นฉบับ
- ห้ามใส่ deadline เอง (ถ้าไม่มีในต้นฉบับ ให้เขียน "TBD")
- ห้ามตัดสินใจแทนทีม — แค่จัดให้เป็นรูปแบบ
- ใช้ official Skill Creator
- ห้ามสร้างไฟล์จนกว่าจะอนุมัติ
```

วิธีใช้ Skill นี้กับทีมจริง

1. หลัง meeting จบ **copy note** หรือ transcript ลงในไฟล์ `meeting-2026-05-24.md`
2. เปิด Claude Code ในโฟลเดอร์ที่มี Skill นี้ → ขอให้รัน Skill กับไฟล์ที่เพิ่งสร้าง
3. Claude สร้าง `output/2026-05-24/` มีไฟล์ทั้ง 7 ตัว
4. **คุณตรวจ** action items + deadline ก่อน — ถ้ามีอะไรขาดให้ Claude ถาม
5. Copy email + chat message ไปส่งจริง — ใช้เวลาทั้งหมด 5 นาที หลัง meeting

🎯 **เพิ่ม value ด้วย project context**

ใส่ไฟล์ `references/project-context.md` ที่บอก Claude ว่า — ชื่อโปรเจกต์, ชื่อคนในทีม, ศัพท์เฉพาะ — Skill จะ **เข้าใจ shorthand** เช่น “Jay จัดให้ตามที่คุยกันใน sprint ที่แล้ว” จะรู้ว่า Jay คือใคร และ sprint ไหน

สรุปบทนี้

- Skill ที่ **ทุกทีมต้องใช้** — meeting → action plan ภายใน 5 นาที
- ไม่มี script — ใช้ได้ทันที ไม่ต้องตั้ง dependency
- กฎทอง: **“ห้ามแต่ง action / deadline ที่ไม่มีในต้นฉบับ”**
- เพิ่ม project-context.md ให้ Skill เข้าใจ shorthand และศัพท์ทีม

KEY TAKEAWAY

Skill ที่ดีที่สุดสำหรับทีม = Skill ที่ **ลดงานที่ไม่อยากทำ** ออกได้มากที่สุด ไม่มีใครชอบเขียน follow-up email หลังประชุม — ลบงานนี้ออกจาก workflow แล้วทีมจะรักคุณ

→ บทต่อไป
บทที่ 09 · Research-to-Carousel Planner
พลิกหน้า →

บทที่ เก้า · CHAPTER NINE

09

Research to-Carousel Planner

Skill สำหรับ **educational creator** ที่ทำ carousel post — รับ link / article / report แล้วได้โครง slide พร้อม source notes

Skill นี้มี **กฎจริยธรรมแน่น** — แยก fact จาก interpretation ห้ามแต่ง quote ห้ามให้ผู้อ่านเข้าใจผิดว่าข้อมูลมาจาก source ที่ไม่มีจริง

- × ไม่แต่ง quote / สถิติ — ใช้เฉพาะที่อยู่ใน source
- × ไม่ปนความเห็นกับข้อเท็จจริง — แยก fact vs interpretation ชัด
- × ไม่ใช้ source ที่ paywalled — ใช้เฉพาะที่ผู้ใช้มีสิทธิ์เข้าถึง

Research-to-Carousel — Brief 7 ช่อง

01. ทำอะไร	แปลง source/article/report → โครง carousel post 7-10 slide พร้อม source notes
02. ใช้เมื่อไหร่	ผู้ใช้ส่ง link/note และขอ "carousel" / "educational post" / "แตกสไลด์"
03. Input	1-3 sources (URL / PDF / text) + topic + target audience
04. Output	main insight + slide outline (7-10) + headlines + caption + CTA + source notes + visual direction per slide
05. Workflow	(1) อ่าน sources (2) extract main insight (3) outline slides (4) draft headlines (5) attach source per claim
06. ห้ามทำ	ห้ามแต่ง quote · ห้าม fabricate stats · ห้ามปน fact + opinion · ถ้า source ไม่พอ ขอเพิ่ม
07. ตัวอย่าง	carousel pack 2 ชุดจาก research papers public (CC-BY) + sample visual direction

กฎเหล็ก: แยก Fact จาก Interpretation

นี่คือ Skill ที่ทำพลาดง่ายที่สุดถ้า prompt ไม่ดี — Claude อาจปน ความเห็นกับข้อเท็จจริงจน reader แยกไม่ออก กฎเหล็กคือ **label ทุก claim**

F

Fact

มี source อ้างอิงตรง · มี citation · ตัวเลข/quote ตรงตามต้นฉบับ · Slide ที่มี F → ต้องใส่ source notes

I

Interpretation

ความเห็น / การสรุป / การ extrapolate · ต้อง label "I (ความเห็นของผู้เขียน)" · ห้ามแอบเป็น fact

⚠️ **ตัวอย่างที่ทำพลาด**

Slide: **“72% ของ Gen Z อยากเรียนรู้ AI”** — ดูเป็น fact แต่ถ้า source บอกแค่ **“ผู้ตอบ survey ในเขตเมือง 1,200 คน”** → ต้อง label ว่า **F (จาก survey เฉพาะกลุ่ม)** ไม่ใช่ใช้คำ **“Gen Z”** เหมือนเป็นทั้งหมด

Prompt — สร้าง research-to-carousel-planner

prompt · สร้าง Skill

```
> ช่วยสร้าง Claude Skill ชื่อ research-to-carousel-planner

Skill นี้ใช้แปลง official sources / article / report
ให้เป็นโครง carousel ภาษาไทย

Output ต้องมี:
1. main insight (1 ประโยค)
2. slide outline 7-10 slides
3. headline 5 แบบ
4. caption
5. CTA
6. source notes (ทุก claim ต้องมี source ID)
7. visual direction per slide (ไม่ใช่ภาพจริง)

กติกา:
- แยก fact กับ interpretation โดย label F / I ทุก claim
- ห้ามแต่งข้อมูล quote หรือ stats
- ถ้า source ไม่พอ ขอเพิ่มก่อน
- ใช้ official Skill Creator
- ห้ามสร้างไฟล์จนกว่าจะอนุมัติ
```

ตัวอย่าง output ที่ดี

ตัวอย่าง **main insight** + **slide outline** ที่ Skill นี้ควรส่งกลับ — ใช้เป็น quality bar เวลาทดสอบ

💎 Main Insight (ตัวอย่าง)

F: 4 จาก 5 ของบริษัท SaaS ที่ใช้ AI assistant ใน support ลด ticket time ลง 40% (source: SaaSReport 2025) **I:** บริษัทเล็กสามารถได้ผลคล้ายกันถ้าใช้ Skill ง่าย ๆ ไม่ต้องลงทุนสูง

🎯 Slide Outline (3 slides แรก)

1. **Hook** · 4 ใน 5 ของ SaaS ใหญ่ใช้ AI assistant ตั้งแต่ปี 2024 **(F · SaaSReport 2025)**
2. **Why now** · ราคา AI tool ลดลง 60% ตั้งแต่ 2023 **(F · CB Insights)**
3. **What changes** · ticket time ลด 40% ทำให้ทีม support เน้นงานยาก **(F · SaaSReport)**

สรุปบทนี้

- Skill นี้สำหรับ **educational creator** ที่อยากทำ carousel จาก source จริง
- กฎเหล็ก: **แยก F / I** ทุก claim — ป้องกัน misinformation
- ถ้า source ไม่พอ ต้อง **ขอเพิ่มก่อน** ห้ามแต่ง
- Output ที่ดีต้องมี **source ID** ติดทุก claim

KEY TAKEAWAY

Educational content ที่มีจริยธรรม **สร้างความเชื่อใจ** ได้ยาวกว่า viral content ที่ไม่มี source — Skill นี้บังคับให้คุณรักษามาตรฐาน ตั้งแต่ slide แรก ทำให้คุณกลายเป็น authority ในวงการอย่างเป็นธรรมชาติ

→ บทต่อไป
บทที่ 10 · Personal SOP Builder
พลิกหน้า →

บทที่ สิบ · CHAPTER TEN

10

Personal SOP Builder

Skill ที่ **ช่วยให้คุณ systemize งานตัวเอง** — คุยกับ Claude เพื่อ extract workflow ในหัวคุณ แล้วเขียนเป็น SOP / checklist / prompt template ที่ใช้ซ้ำได้

Skill ปิดของ Part II — เหมาะกับ freelancer / agency / team lead ที่อยากให้ความรู้ในหัวกลายเป็น **“ทรัพย์สิน”** ที่ส่งต่อให้คนอื่นได้

- ✕ ไม่ใช่ SOP ขั้นเทพแบบ enterprise — เริ่มจาก SOP เล็ก ๆ ก่อน
- ✕ ไม่บังคับ format เดียว — ปรับให้เข้ากับงานคุณได้
- ✕ ไม่ใช่เครื่องมือทำ org-wide policy — เน้น personal workflow

Personal SOP Builder — Brief 7 ช่อง

01. ทำอะไร	สัมภาษณ์ผู้ใช้เรื่อง workflow ที่ทำซ้ำ → เขียนเป็น SOP + checklist + prompt template
02. ใช้เมื่อไหร่	ผู้ใช้บอก "ฉันทำงาน X ซ้ำๆ อยากเขียน SOP" / "systemize workflow ฉันที"
03. Input	ชื่องาน + ความถี่ + ใครทำ + ปัญหาที่เจอบ่อย
04. Output	SOP ทีละขั้น + checklist + prompt template + common mistakes + quality bar + handoff instruction
05. Workflow	(1) ถามทีละ 3 คำถาม (2) สรุปสิ่งที่เข้าใจ (3) เขียน SOP v1 (4) ผู้ใช้ revise (5) เก็บ final
06. ห้ามทำ	ห้ามถามยาว ๆ ในรอบเดียว · ห้ามแต่งขั้นตอนที่ผู้ใช้ไม่ระบุ · ห้ามใส่ tool ที่ผู้ใช้ไม่มี
07. ตัวอย่าง	SOP ตัวอย่าง 2 ชุด: (1) blog post review (2) client onboarding

Output 6 ส่วนของ SOP ที่ใช้งานได้จริง

01

SOP ทีละขั้น

ขั้นตอนทำงานละเอียดพอที่คนใหม่ทำตามได้ ไม่ต้องเดา

02

Checklist ก่อนส่ง

รายการตรวจก่อน deliver — ทำให้คุณภาพไม่ตกตอนเร่ง

03

Prompt Template

prompt พร้อมใช้สำหรับ Claude เวลาทำงานนี้ครั้งถัดไป

04

Common Mistakes

สิ่งที่ผู้ใช้เคยพลาด — ช่วยคนใหม่หลีกเลี่ยง

05

Quality Bar

เกณฑ์วัดว่า output "ดีพอจะส่ง" — ลด over-polish

06

Handoff Instruction

คำแนะนำเมื่อต้องส่งงานให้คนอื่นทำต่อ — ลด rework

ทำไม Skill นี้เหมาะกับ freelancer

คุณเก็บ SOP แต่ละงาน → ใช้ Skill เดียวกันสร้าง SOP งานอื่น → หลัง 3-6 เดือน คุณมี **“SOP library ส่วนตัว”** ที่ขายเป็น digital product ได้ หรือใช้ตอน hire ฟรีแลนซ์รุ่นน้อง

Prompt — สร้าง personal-sop-builder

prompt · สร้าง Skill

```
> ช่วยสร้าง Claude Skill ชื่อ personal-sop-builder

Skill นี้ช่วยสัมภาษณ์ฉันเกี่ยวกับ workflow ที่ฉันทำซ้ำ ๆ
แล้วแปลงเป็น SOP ที่ใช้งานได้จริง

Output ต้องมี:
- SOP ทีละขั้น
- checklist ก่อนส่งงาน
- prompt template สำหรับใช้ซ้ำ
- common mistakes
- quality bar
- handoff instruction ให้คนอื่นทำต่อได้

กติกา:
- ถ้าฉันอธิบายไม่ครบ ถามทีละ 3 คำถาม
  ไม่ถามยาวเกิน
- ห้ามแต่งขั้นตอนที่ฉันไม่ได้ระบุ
- ห้ามใส่ tool ที่ฉันยังไม่มี
- สรุปสิ่งที่เข้าใจกลับให้ฉันก่อนเขียน SOP v1
- ใช้ official Skill Creator
- ห้ามสร้างไฟล์จนกว่าจะอนุมัติ
```

วิธีใช้ Skill นี้กับชีวิตจริง

1. เลือก workflow 1 อย่างที่คุณทำซ้ำสัปดาห์ละครั้งขึ้นไป
2. เปิด Claude Code → รัน Skill นี้ → Claude เริ่มถาม
3. ตอบทีละชุด — ถ้าตอบไม่ได้ก็บอก "ขอคิดก่อน" Claude รอ
4. หลังถามเสร็จ Claude สรุปกลับ → คุณ revise สิ่งที่เข้าใจผิด
5. เก็บ SOP ใน `my-sops/[workflow-name].md`
6. เดือนหน้า — ใช้ SOP จริง พบจุดที่ขาด ปรับเป็น v2

🎯 **ตัวอย่าง workflow ที่ควรทำ SOP ก่อน**

- **Onboarding ลูกค้าใหม่** — ทุก agency มี repeat process
- **Review blog post ก่อนเผยแพร่** — มี standard ในหัวอยู่แล้ว
- **ส่งงานสิ้นเดือนให้ client** — มี checklist อยู่แล้วในใจ
- **ตัดสินใจรับงานหรือไม่** — ใช้ filter ซ้ำ ๆ

สรุปบทนี้

- Personal SOP Builder = Skill ที่เปลี่ยน **ความรู้ในหัว** เป็น **ทรัพย์สิน**
- Output 6 ส่วน: SOP / checklist / prompt template / mistakes / quality bar / handoff
- Claude สัมภาษณ์ทีละ 3 คำถาม ไม่ถามยาว — ทำให้คุณตอบได้ครบไม่เหนื่อย
- หลัง 3-6 เดือน คุณจะมี **SOP library** ที่ขายได้

KEY TAKEAWAY

Skill ที่ดีที่สุดสำหรับคุณ **ในระยะยาว** อาจไม่ใช่ Skill ที่ขายดี แต่เป็น Skill ที่ **"ทำให้คุณทำงานเป็นระบบ"** เพราะมันต่อยอดเป็น Skill อื่น ๆ ได้ไม่จำกัด — เริ่มจาก workflow เดียวก่อน แล้วขยาย

→ บทต่อไป · เริ่ม PART III
บทที่ 11 · วิธีทดสอบ Skill แบบคนไม่รู้โค้ด
พลิกหน้า →

ส่วนที่ 3

III

ทดสอบและแพ็ก

สร้าง Skill เสร็จยังไม่จบ ต้องทดสอบกับงานจริงและตรวจก่อนแพ็กไปใช้กับคนอื่น

บทที่ สิบเอ็ด · CHAPTER ELEVEN

11

วิธีทดสอบ Skill แบบคนไม่รู้โค้ด

Skill ที่สร้างเสร็จ **ยังไม่ใช่ Skill ที่ใช้งานได้** — ต้องผ่านการทดสอบกับงานจริงก่อนถึงจะมั่นใจ

บทนี้สอนวิธี **ทดสอบ Skill 3 case** และให้ Claude **grade output ตามเกณฑ์** เพื่อหาจุดที่ต้องปรับ — ไม่ต้องเขียน test code เอง

- ✕ ไม่ใช่ unit test แบบโปรแกรมเมอร์ — เป็น manual test แบบใช้งานจริง
- ✕ ไม่ใช่การหา bug ทางเทคนิค — หาความ "ไม่ตรงงาน"
- ✕ ไม่ต้องเขียน assertion หรือ Pytest — Claude grade ให้

Test case คืออะไรแบบง่าย

Test case = **input หนึ่งชุด + output ที่คาดหวัง** ทดสอบ Skill = ส่ง input เข้าไป → ดู output → ถามตัวเองว่าตรงที่คาดไหม

01

Case ง่าย

input ที่ Skill น่าจะตอบได้ดีที่สุด — "happy path" · เช่น input ครบ tone ชัด งานคุ้นเคย

02

Case ใช้จริง

งานจริงในชีวิตประจำวันของคุณ — input แบบที่คุณจะส่งจริงสัปดาห์หน้า

03

Case กวน

input ที่ตั้งใจให้ "ขาด" — Skill ต้อง **ถามกลับ** ไม่ใช่เดา

⚠️ **อย่าทดสอบแค่ case ง่าย**

ความผิดพลาดอันดับ 1 ของมือใหม่ — ทดสอบ case ง่ายแล้วเห็นว่าใช้ได้ ก็คิดว่า Skill เสร็จแล้ว ปรากฏว่าใช้งานจริงพังเพราะ case จริงไม่ใช่ "ง่าย" ทุกครั้ง **case กวน** คือตัวจริงที่บอกว่า Skill นี้ robust หรือไม่

เกณฑ์ Grade output 4 ข้อ

ใช้ 4 เกณฑ์นี้เป็น checklist — ครบทุกข้อ = Skill ผ่าน ขาดข้อใดข้อหนึ่ง = ต้องปรับ SKILL.md / description / workflow

01

ทำตามหน้าที่ Skill ไหม

Output ตรงกับสิ่งที่ Skill brief บอกไหม · มีครบทุก section หรือเปล่า · format ตรงไหม

02

ถามกลับเมื่อข้อมูลไม่พอไหม

ถ้า input ขาด Skill ต้องถาม ไม่ใช่เดา · เป็นเกณฑ์สำคัญสำหรับ case กวน

03

Output ใช้งานจริงไหม

Copy ไปใช้ได้เลยไหม · ต้องแก้เยอะแค่ไหนก่อน publish · ถ้าแก้เยอะ = Skill ยังไม่พร้อม

04

มีส่วน "น้ำ" ไหม

ประโยคไหนตัดทิ้งได้โดยไม่เสีย value · ถ้ามีน้ำมาก = description ต้องเข้มขึ้น

Prompt — ให้ Claude ทดสอบและ grade Skill

Prompt นี้ใช้หลังสร้าง Skill เสร็จ — ให้ Claude สร้าง test case ทดสอบ Skill ของคุณ และเสนอวิธีปรับ

prompt · ทดสอบ + grade + เสนอแก้

```
> ช่วยทดสอบ Skill [ชื่อ Skill] ด้วย test cases 3 แบบ:

1. case ง่าย — input ที่ตรงกับสิ่งที่ Skill ออกแบบมา
2. case งานจริง — input แบบที่ฉันจะส่งจริงในชีวิตประจำวัน
3. case กวน — input ที่ขาดข้อมูล / ผิด format / ขัดกับ trigger

ให้ประเมิน output แต่ละ case ตามเกณฑ์:
- ทำตามหน้าที่ Skill ไหม
- ถามกลับเมื่อข้อมูลไม่พอไหม
- output ใช้งานจริงไหม
- มีส่วนไหนยาว/น้ำ/ไม่จำเป็นไหม

จากนั้น เสนอวิธีปรับ SKILL.md ให้ดีขึ้น
ก่อนแก้ ขออนุมัติฉันก่อน
```

วนลูป improve — เก่งขึ้นทีละนิด

1. ทดสอบ 3 case → grade ทั้งหมด
2. ถ้าผ่าน 4/4 ทุก case → Skill **พร้อมใช้งาน**
3. ถ้าไม่ผ่าน ระบุข้อที่ตก → ปรับ SKILL.md / description / workflow
4. ทดสอบใหม่อีกรอบ → grade
5. วน **2-3 รอบ** โดยปกติพอ — ถ้าเกิน 5 รอบ = brief ยังไม่ชัด ให้กลับไปที่ ch04

💡 **เก็บ test cases ไว้สำหรับอนาคต**

เก็บ test cases ที่ใช้ไว้ใน `tests/cases.md` ในโฟลเดอร์ Skill — เวลาคุณ update Skill ในอนาคต ใช้ test cases เดิม regress test ได้ ป้องกัน Skill ใหม่ทำ case เดิมพัง

🎯 **Skill ที่ผ่าน test แล้วยังไงต่อ**

1. ใช้งานจริง 1-2 สัปดาห์
2. ถ้ามีจุดที่ทำพลาด → กลับมาเพิ่ม test case + ปรับ Skill
3. เมื่อมั่นใจ → แพ็กและแชร์ (บทที่ 12)

สรุปบทนี้

- Test ด้วย **3 case**: ง่าย · งานจริง · กวน — แต่ละ case test คนละด้าน
- Grade ด้วย **4 เกณฑ์**: หน้าที่ · ถามกลับ · ใช้จริง · มีน้ำไหม
- ให้ Claude **เสนอแก้** ไม่ใช่แก้เลย — คุณอนุมัติก่อน
- เก็บ test cases ไว้ regress test ทุกครั้งที่ update Skill

KEY TAKEAWAY

Skill ที่ผ่านการทดสอบ **ใช้งานได้ทุกครั้ง** — Skill ที่ไม่ทดสอบ **ใช้ครั้งสองครั้งดี ครั้งที่สามพัง** ลงทุน 30 นาทีทดสอบหลังสร้าง = ประหยัดความหงุดหงิดเป็นเดือน ๆ

→ บทต่อไป · บทสุดท้ายของเล่ม
บทที่ 12 · แพ็ก แชร์ และใช้ Skill ซ้ำ
พลิกหน้า →

บทที่ สิบสอง · CHAPTER TWELVE

12

แพ็ก แชร์ และใช้ Skill ซ้ำ

Skill ที่สร้างเสร็จและทดสอบผ่านแล้ว — จะใช้คนเดียวก็ได้ หรือ **แพ็กแชร์ทีม / ขายเป็น digital product** ก็ได้

บทสุดท้ายของเล่ม — สอนวิธีตรวจไฟล์ก่อนแชร์ (สำคัญที่สุด!) ทำ README ทำ skill pack และคิด version แรก

- ✕ ไม่ใช่บทสอนทำ landing page — เน้นการแพ็ก Skill ให้พร้อมขาย
- ✕ ไม่ใช่บทสอน marketing — แค่บอกว่าต้องมีอะไรใน package
- ✕ ไม่ใช่บทสุดท้ายเรื่อง Skill — มี ภาคผนวก A/B/C ต่อ

ตรวจไฟล์ก่อนแชร์ — กฎความปลอดภัย

🚨 **อย่าใส่ secret ใน Skill folder**

API key, password, session token, OAuth client secret, personal email, ลูกค้าจริงที่ไม่ได้ขอ permission — **ห้ามใส่**ใน SKILL.md / references / scripts เคยมีคน push token Slack ขึ้น GitHub ผ่าน Skill folder = ทีมพังภายในวัน

Checklist ก่อนแชร์ Skill

1. เปิดทุกไฟล์ในโฟลเดอร์ → ไม่มีอะไรที่ขึ้นต้นด้วย `sk- ...` หรือ `token=`
2. ตรวจ references — ไม่มี client name / project ของจริงที่ไม่ได้ขอ permission
3. ตรวจ assets — ไฟล์ที่มีจริงมีลิขสิทธิ์ใช้แชร์ไหม
4. ตรวจ scripts — ไม่มี hardcoded path ที่ชี้ไปยังเครื่องคุณ (`/Users/you/ ...`)
5. ทดสอบรัน Skill ใน folder ใหม่ที่ไม่มี context เดิม → ใช้งานได้ไหม

💡 **Prompt ช่วยตรวจ**

ใช้ prompt นี้กับ Claude Code ในโฟลเดอร์ Skill — มันจะ scan ให้

```
ช่วยตรวจ Skill folder นี้ก่อนแชร์ — หา secret, hardcoded path, ชื่อลูกค้าจริง, ไฟล์ลิขสิทธิ์ที่อาจมีปัญหา
```

README สั้น ๆ สำหรับคนใช้

Skill ที่แชร์ต้องมี README — คนใช้รู้ทันทีว่า Skill นี้คืออะไร / ใช้ยังไง ไม่ต้องเปิด SKILL.md เพื่อเข้าใจ

📄

README โครงพื้นฐาน

(1) Skill นี้ทำอะไร (2) ใช้เมื่อไหร่ (3) วิธีติดตั้ง (4) ตัวอย่างใช้งาน 1-2 (5) ข้อจำกัด

🔢

Version แรก = 0.1.0

อย่าตั้ง 1.0.0 — บอกผู้ใช้ว่ายังเป็น beta · ใช้ semver: 0.1 → 0.2 (เพิ่ม feature) → 1.0 (production-ready)

Skill สำหรับใช้เอง vs แจกทีม vs ขาย

ใช้เอง

เก็บใน Dropbox/iCloud · ไม่มี README ก็ได้ · ไม่ต้องระวัง secret มาก

แจกทีม

GitHub private / Slack · ต้องมี README · ตรวจ secret · มี handoff doc

ขาย

ต้องครบ: README · examples · troubleshooting · usage guide · sales page · refund policy

Skill Pack สำหรับขาย — ต้องมี 5 อย่าง

01

Skill Folder

Skill หลัก (อาจมีหลายตัวรวมเป็น pack) · ผ่าน test แล้ว · ตรวจ secret แล้ว

02

Usage Guide

PDF / Notion 5-10 หน้า สอนติดตั้งและใช้งาน · มีภาพประกอบ

03

Example Prompts

10-20 prompt copy-ready สำหรับงานหลากหลาย · เป็นจุดที่ลูกค้ารัก

04

Sample Input/Output

ตัวอย่างจริง 3-5 ชุด ให้เห็นว่าคุณภาพ output เป็นยังไง

📦 **Troubleshooting (อย่าลืม)**

หน้า troubleshooting — คำถามที่ลูกค้าน่าจะถาม + คำตอบสั้น ลด support load 80% ปกติ 10-15 คำถามก็พอ · ใส่ความเป็นมิตรไม่ใช่ FAQ แห้ง ๆ

Prompt — ให้ Claude ทำ package ครบเซ็ต

prompt · ตรวจและแพ็ก Skill

```
> ช่วยตรวจ Skill folder นี้ก่อนนำไปแชร์

ตรวจให้หน่อยว่า:
- มีไฟล์ลับหรือข้อมูลส่วนตัวไหม
- instructions ชัดไหม
- description trigger ดีไหม
- มีตัวอย่างการใช้งานไหม
- คนไม่รู้โค้ดจะใช้ได้ไหม

จากนั้นช่วยทำ:
1. README ภาษาไทย สำหรับผู้ใช้ทั่วไป
2. Usage Guide 5-7 หน้า มีภาพประกอบ (ระบุจุด)
3. Example Prompts 10 ชุด
4. Troubleshooting 10 ข้อ
5. Sample Input/Output 3 ชุด

ใส่ทุกอย่างใน folder ใหม่ชื่อ dist/
แยกจาก Skill folder ต้นฉบับ
```

สรุปบทสุดท้าย — และปิดเล่ม

- ก่อนแชร์ — **ตรวจ secret** เสมอ (กฎที่สำคัญที่สุดของบทนี้)
- **README + version** — บอกผู้ใช้ว่ามันคืออะไร อยู่ stage ไหน
- Skill ที่จะขาย ต้องมี **5 อย่าง**: folder · usage guide · prompts · samples · troubleshooting
- ใช้ Claude ช่วยตรวจและทำ package — ใช้เวลาไม่กี่ชั่วโมง

ปิดเล่ม

คุณได้เดินทางจาก **“Skill คืออะไร”** ไปจนถึง **“แพ็ก Skill ขาย”** ผ่าน 12 บท + 6 Skill ตัวอย่าง
เลือก Skill ตัวที่ใกล้กับงานคุณที่สุด — สร้างมันคืนนี้ ทดสอบพรุ่งนี้ ใช้งานสัปดาห์หน้า อย่ารอความสมบูรณ์แบบ Skill ตัวแรกที่ **“พอใช้ได้”** ดีกว่า Skill ตัวที่ **“ดีที่สุด”** ที่ยังไม่เริ่ม

→ ต่อไป
ภาคผนวก · Prompt พร้อมใช้ · คำศัพท์ · Source Map
คู่มืออ้างอิง →

ภาคผนวก

คู่มืออ้างอิงเร็ว

Prompt พร้อมใช้ · คำศัพท์ที่ต้องรู้ · Official Sources Map

ภาคผนวก A · CHAPTER A

A

Prompt สั้นพร้อมใช้ 10 ชุด

รวม prompt ทั้งหมดที่ใช้ในเล่ม + prompt เพิ่มเติมที่ใช้ซ้ำบ่อย **copy-ready** — ไม่ต้องคิดเอง

เก็บ ภาคผนวก A นี้ **bookmark ไว้** เปิดทุกครั้งที่จะเริ่มงานใหม่กับ Claude

Prompt 1–3 · เริ่มต้นและทำความเข้าใจ

1. ให้ Claude อธิบาย Skill แบบคนทั่วไป

```
> ช่วยอธิบาย Claude Skill ให้ฟังหน่อย
แบบที่คุยกับคนที่ใช้ AI เป็นแต่ไม่รู้โค้ด

กติกา: ภาษาไทย คำง่าย ไม่เกิน 6 บรรทัด
ปิดท้ายด้วยคำถามว่าฉันอยากเริ่มทำ Skill อะไรก่อน
```

2. ตรวจเครื่องครั้งแรก

```
> ฉันกำลังจะสร้าง Claude Skill แต่ฉันไม่รู้โค้ด
ช่วยตรวจเครื่องนี้ให้หน่อยว่า Claude Code พร้อมใช้งานไหม

- อธิบายภาษาไทยสั้น ๆ
- อย่ารันคำสั่งอันตราย
- ถ้าต้องติดตั้ง ใช้ official เท่านั้น และขออนุมัติก่อน
```

3. อธิบายโครงสร้าง Skill จาก official spec

```
> ช่วยอธิบายโครงสร้าง Claude Skill จาก official spec
แบบคนไม่รู้โค้ด ขอเฉพาะส่วนที่ต้องรู้จริง

อ่าน: github.com/anthropics/skills/blob/main/skills/skill-
creator/SKILL.md
สรุปไม่เกิน 8 บรรทัด
```

Prompt 4–6 · สร้าง Skill

4. เปลี่ยนไอเดียเป็น Skill Brief 7 ช่อง

```
> ช่วยเปลี่ยนไอเดียนี้ให้เป็น Skill Brief 7 ช่อง:

ไอเดีย: [ใส่ไอเดีย]

ขอ output 7 ส่วน:
ทำอะไร / ใช้เมื่อไหร่ / Input / Output /
Workflow / ห้ามทำ / ตัวอย่าง

ถ้าไอเดียไม่ชัด ถามฉันก่อนทีละ 2-3 คำถาม
```

5. สร้าง Skill ที่ไม่มี script

```
> ช่วยสร้าง Claude Skill ชื่อ [name]
ตาม Skill Brief ที่ฉันแนบมา

โปรดอ่าน Skill Creator ก่อน:
- claude.com/plugins/skill-creator
- github.com/anthropics/skills/blob/main/skills/skill-creator/SKILL.md

ฉันไม่รู้โค้ด · ห้ามสร้างไฟล์จนกว่าจะอนุมัติ
```

6. สร้าง Skill ที่มี script (ต้องติดตั้ง tool)

```
> ช่วยสร้าง Claude Skill ชื่อ [name] ที่ต้องใช้ tool: [list]

ก่อนติดตั้ง tool ขออนุมัติฉันก่อนทุกครั้ง
ใช้ official source สำหรับติดตั้ง

ฉันไม่รู้โค้ด · อธิบายภาษาไทย ทีละขั้น
```

Prompt 7–8 · ทดสอบและปรับ

7. ทดสอบ Skill ด้วย 3 case + grade

```
> ช่วยทดสอบ Skill [name] ด้วย 3 test case:

1. case ง่าย — input ที่ Skill ออกแบบมา
2. case จริง — input ที่ฉันส่งจริงในชีวิตประจำวัน
3. case กวน — input ที่ขาดข้อมูล / ผิด format

ประเมินตามเกณฑ์:
- ทำตามหน้าที่ไหม
- ถามกลับเมื่อข้อมูลไม่พอไหม
- output ใช้งานจริงไหม
- มีส่วนน้ำไหม

เสนอแก้ SKILL.md — ขออนุมัติก่อนแก้
```

8. ปรับ description เพื่อ trigger ดีขึ้น

```
> Skill [name] นี้ Claude ไม่หยิบมาใช้ตอนที่ควรหยิบ

ช่วยวิเคราะห์ description ใน SKILL.md
แล้วเสนอ description ใหม่ที่ trigger ตรงกว่า

อธิบายเหตุผลที่ปรับ
ขออนุมัติก่อนแก้ไฟล์
```

Prompt 9–10 · แพ็กและขาย

9. ตรวจ Skill folder ก่อนแชร์

```
> ช่วยตรวจ Skill folder นี้ก่อนนำไปแชร์

หา:
- secret / API key / token / password
- hardcoded path ที่ชี้ไปยังเครื่องฉัน
- ชื่อลูกค้าจริงที่ฉันไม่ได้ขอ permission
- ไฟล์ที่อาจมีปัญหาลิขสิทธิ์

จากนั้นทำ README ภาษาไทยสำหรับผู้ใช้ทั่วไป
```

10. ทำ skill pack สำหรับขาย

```
> ช่วยทำ skill pack สำหรับขาย จาก Skill folder นี้

ใส่ใน folder dist/:
1. README ภาษาไทย
2. Usage Guide 5-7 หน้า
3. Example Prompts 10 ชุด
4. Sample Input/Output 3 ชุด
5. Troubleshooting 10 ข้อ

อย่าใส่ secret · ภาษาเป็นมิตร ไม่ใช่ FAQ แห้ง ๆ
```

Bonus prompt

ใช้ prompt เหล่านี้ **เป็นจุดเริ่ม** — ปรับให้เข้ากับงานคุณได้ ใส่บริบทเพิ่ม เช่น “ตอนนี้กำลังทำ Skill สำหรับ [งาน]” หรือ “ลูกค้าหลักของฉันคือ [กลุ่ม]” ผลลัพธ์จะดีขึ้นเยอะ

ภาคผนวก B · CHAPTER B

B

คำศัพท์ที่ต้องรู้

Glossary แบบ **ภาษาคนทั่วไป** — ไม่ใช้ศัพท์เทคนิคซ้อนศัพท์เทคนิค

เปิดอ่านเมื่อเจอคำที่ไม่แน่ใจในเล่ม ทุกคำมี **“1 ประโยคพอใช้งาน”** และ **“ทำไมต้องรู้”**

คำศัพท์หลัก

Skill	ชุดคำสั่ง + เครื่องมือย่อยที่ Claude หยิบมาใช้เมื่อเจองานแบบเดียวกัน · เก็บเป็นไฟล์ในเครื่อง
Agent Skill	ชื่อทางการของ Skill ใน Anthropic spec · ใช้สลับกันได้กับคำว่า "Skill"
SKILL.md	ไฟล์หัวใจของ Skill ทุกตัว · บอกว่า Skill นี้คืออะไร · ใช้เมื่อไหร่ · ทำอะไร
description	ฟิลด์ใน SKILL.md ที่บอก Claude ว่า trigger ของ Skill นี้คือเมื่อไหร่ · สำคัญที่สุด
trigger	สัญญาณที่ทำให้ Claude รู้ว่าควรเรียก Skill นี้ · เกิดจาก description ที่ชัดพอ
instructions	ส่วนใน SKILL.md ที่บอก Claude ว่าต้องทำอะไรเมื่อ Skill ทำงาน · ขั้นตอน workflow

คำเทคนิคที่ต้องรู้ (พอใช้งาน)

script	ไฟล์ Python / shell ที่ Skill เรียกใช้สำหรับงานที่ต้องประมวลผล (เช่น แตก video)
reference	ไฟล์เอกสารใน references/ ที่ Claude อ่านเป็น context (brand guide, glossary)
asset	ไฟล์ที่ Skill ใช้แสดง/อ้างอิง (รูป, template, sample) เก็บใน assets/
CLI	Command Line Interface · ช่องดำในเครื่องที่พิมพ์คำสั่งแทนการคลิก
Terminal	App สำหรับใช้ CLI · macOS = Terminal · Windows = PowerShell · Linux = หลายตัว
official source	เว็บไซต์/repo ทางการของผู้สร้าง tool · เช่น claude.com ไม่ใช่ claude.ai-something.com
dependency	tool / library ที่ Skill ต้องใช้เพิ่ม · เช่น yt-dlp, ffmpeg ใน ch06
package	1. ชุดไฟล์ที่ติดตั้งผ่าน npm/pip · 2. Skill pack = ชุด Skill หลายตัวรวมเป็นของขาย

คำที่ใช้ในการทดสอบและแพ็ก

test case	input หนึ่งชุดสำหรับทดสอบ Skill · มี 3 แบบ: ง่าย / จริง / กวน (บทที่ 11)
eval	การประเมิน output ของ Skill ตามเกณฑ์ · ในเล่มนี้ใช้ 4 เกณฑ์
regress test	ทดสอบใหม่ด้วย case เดิม · ป้องกัน Skill ใหม่ทำงานเดิมพัง
version	เลขรุ่น Skill เช่น 0.1.0 · semver: major.minor.patch
semver	ระบบเลข version · เพิ่ม patch (0.1.0 → 0.1.1) เมื่อแก้ bug · เพิ่ม minor เมื่อเพิ่ม feature
skill pack	ชุด Skill 3-5 ตัวรวมเป็นของขาย · เพิ่ม value มากกว่าขาย Skill ทีละตัว
README	ไฟล์เริ่มต้นที่คนใช้เปิดอันแรก · บอกว่า Skill นี้คืออะไร / ใช้ยังไง
Skill Brief	สูตร 7 ช่องสำหรับออกแบบ Skill ก่อนสร้าง (บทที่ 4) · ลด rework ได้ครึ่งหนึ่ง

ภาคผนวก C · CHAPTER C

C

Official Sources Map

รวม source ทางการที่ใช้อ้างอิงในเล่ม — เปิดเพื่อตรวจสอบเอง / อ่านต่อ / ส่งให้ Claude อ่านในอนาคต

ทุก URL ในหน้านี้คือ **official source** จาก Anthropic หรือ project ของ tool ที่ใช้ในเล่ม ห้ามใช้ source อื่นที่ไม่ใช่ official

Anthropic / Claude — Official

Claude Code	claude.com/code · install + docs + IDE integration
Skill Creator plugin	claude.com/plugins/skill-creator · plugin page + getting started
Agent Skills repo	github.com/anthropics/skills · official open-source skills
Skill Creator SKILL.md	github.com/anthropics/skills/blob/main/skills/skill-creator/SKILL.md
Agent Skills spec	Claude Platform docs · ค้นด้วยคำว่า "Agent Skills" บน claude.com/docs
Claude Help Center	support.anthropic.com · มี section เรื่อง Skills + Claude Code
Anthropic Engineering	anthropic.com/engineering · blog post เกี่ยวกับ design ของ skill system

⚠ **ตรวจ URL ก่อนคลิกเสมอ**

เว็บปลอมที่ลอก URL Anthropic มีหลายตัว — เช็คให้แน่ใจว่าเป็น **claude.com** หรือ **anthropic.com** เท่านั้น ไม่ใช่ claude-something.com หรือ claude.ai-extension.com

Tools ภายนอกที่ใช้ในเล่ม

yt-dlp (บทที่ 6)	github.com/yt-dlp/yt-dlp · ดาวน์โหลด YouTube · MIT license
ffmpeg (บทที่ 6)	ffmpeg.org · audio/video conversion · LGPL/GPL license
Python	python.org · มาในเครื่อง macOS/Linux แล้ว · Windows ติดตั้งจาก official
Node / npm	nodejs.org · ใช้สำหรับ install Claude Code · LTS version แนะนำ

วิธีให้ Claude research source เอง

ถ้าในอนาคตคุณเจอ tool ใหม่ที่ Skill ของคุณต้องใช้ — อย่าเชื่อ tutorial ของ third party ทันที ใช้ prompt นี้กับ Claude:

💡 **Prompt research tool ใหม่**

ฉันอยากใช้ tool [name] ใน Skill ของฉัน ช่วยเปิด official docs แล้วสรุปว่า (1) มันทำอะไร (2) ติดตั้งยังไง (3) ใช้ใน command line ยังไง (4) มี license อะไร (5) มีปัญหา security ที่ public ไหม ใช้เฉพาะ official source — ห้ามใช้ blog post หรือ tutorial

จบเล่ม — ขอบคุณที่อ่านมาถึงตรงนี้

หนังสือเล่มนี้คือ **จุดเริ่มต้น** — ไม่ใช่จุดจบ Skill ตัวแรกของคุณจะไม่สมบูรณ์ และก็ไม่จำเป็นต้องสมบูรณ์ **สร้างคืนนี้ ทดสอบพรุ่งนี้ ปรับสัปดาห์หน้า ขาย/แจกเดือนถัดไป**