DAY TEN

10 /10

Bringing It Together — *the whole map, and becoming the teacher*

Nine days, many surfaces, one set of habits running through all of them. Today ties the threads into a single working map, sets a capstone that proves it, and turns the learner into someone who can teach this series themselves.

BY THE END OF DAY 10, YOU WILL BE ABLE TO TEACH OTHERS TO —

- Choose the right Claude surface for any job
- Plan and start a real multi-surface project
- Run a cohort through all ten days
- Name the habits that run through every day
- Teach this series to someone else
- Keep their practice current as things change

01 Why today matters

ORIENTATION

Nine days have each added a piece: what Claude is, the chat surfaces, prompting, structure, Claude Code, its practice, the API, connection, and design. Today is not a tenth piece — it is the assembly. The goal is for a learner to walk away holding one coherent map, not ten separate lessons, and to be ready to teach that map to someone else.

This is also the day the series turns outward. Up to now the learner has been a student. Today they become a teacher — and the teaching notes are written for that handover. Treat this as both a capstone and a commencement.

PREREQUISITES

All nine prior days. Today assumes the vocabulary and habits of the whole series and works at the level of synthesis — if a day is shaky, revisit it before teaching this one.

02 The whole map — choosing a surface

CORE CONCEPT

The series covered several surfaces. The skill now is choosing well — matching the job to the right one without hesitating.

> THE DECISION
>
> **Who or what is doing the work?**
>
> One question sorts most cases. If a person is doing the thinking with Claude's help — drafting, analysing, exploring — that is the chat apps. If the work is in a codebase, that is Claude Code. If your software needs Claude inside it, that is the API. If Claude needs to reach your systems, that is connectors and MCP. If you are making a thing with Claude, that is the design partnership.

THE MAP IN ONE PASS

- ✓ **Claude.ai apps** — a person thinking, drafting, analysing (Days 1–4)
- ✓ **Claude Code** — work inside a real codebase (Days 5–6)
- ✓ **The API** — Claude built into your own software (Day 7)
- ✓ **Connectors & MCP** — Claude reaching your tools and data (Day 8)
- ✓ **Design partnership** — making a real, usable thing together (Day 9)

These are not rival products — they are one capability with different doorways. Many real projects use several at once.

03 The habits that ran through everything

CORE CONCEPT

The surfaces change. The habits did not. These six threads appeared on every day of the series — name them, and a learner has the portable core that outlasts any specific feature.

- **Be specific.** From Day 3's brief to Day 9's design direction — vague in, vague out, every time.
- **Give complexity structure.** Day 4's tagged prompts, Day 6's scoped tasks — hard work needs visible shape.
- **Ask for reasoning on hard things.** Plan-first, think-it-through — a fast answer to a hard question is a risky one.

- **Stay in the loop.** Review the diff, approve the step, judge the design. The more Claude can do, the more this matters.
- **Verify against reality.** Ground answers, run the tests, check the output. Confidence is not correctness.
- **Iterate.** First output is a draft. The good result is on round three, steered with intent.

> *"Surfaces are what you learn. Habits are what you keep. Teach the habits hardest."*

04 Becoming the teacher

CORE CONCEPT

This series was always meant to be passed on. Carrying it well rests on a few teaching principles — the same ones the Teaching Notes have modelled for nine days.

4.1 Principles that travel

- ✓ **Open with the gap.** Every day started by making a problem felt before naming the fix. Curiosity first, content second.
- ✓ **Concept before mechanics.** The mental model carries the feature. Teach the idea; the buttons follow.
- ✓ **Hands on every day.** The Lab is not optional. People learn this by doing it, not hearing about it.
- ✓ **Pre-empt the misconception.** Name the wrong model out loud before it sets. It is far cheaper than correcting it later.

4.2 Running a cohort

One day at a time is the design — space to practise between sessions is where it sticks. Keep each session anchored on its Lab. Let the "check for understanding" questions be a real gate, not a formality. And teach the arc, not just the days: learners should feel the through-line from "meet Claude" to "build something real."

> KEY IDEA TO INSTALL
>
> You do not need to be the world's expert to teach this well. You need the map, the habits, and the willingness to learn alongside the room. A teacher who is still learning, openly, is the most credible kind.

05 Habits that compound — and staying current

CORE CONCEPT

The series ends, but the practice does not. A few habits keep a learner — and a teacher — getting better long after Day 10.

- **Reach for the right surface by reflex.** The map in Section 02 should become instinct, not a lookup.
- **Stay specific under pressure.** Specificity is easy when calm and the first thing to go when rushed. Hold it anyway.
- **Keep the human in the loop as capability grows.** As Claude can do more, the discipline of reviewing and directing becomes more important, not less.
- **Teach to learn.** Explaining this to someone else is the fastest way to find the soft spots in your own understanding.
- **Stay current.** The specific features will move. The habits will not. When something changes, check the official documentation and update the detail — the framework holds.

THE ONE HONEST CAVEAT

This guide is a snapshot. Models, limits, and features evolve. That is not a weakness in the series — it is why the series is built on durable habits rather than perishable specifics. Teach the thinking, verify the facts, and the work stays good.

LAB 10 ~50 MIN

The capstone — and your first session

Two parts. First, prove the map by planning a real project that uses more than one surface. Then, prepare to teach by planning your first session.

1. **Choose a real project** of your own — something genuinely worth doing, not a toy.
2. **Map it across surfaces.** Which parts are the apps, Claude Code, the API, connectors, the design partnership? Most real projects touch several.
3. **Name the habits it needs.** Where will specificity, structure, reasoning, the loop, verification, and iteration each have to show up?
4. **Start it.** Take the first concrete step today — the first prompt, the first session, the first sketch.
5. **Plan your first teaching session.** Pick the day you would teach first, write its opening "gap," and decide how you would run its Lab. You are the teacher now.

a real project underway across the right surfaces — and a learner who has stopped being a student and started being a teacher, with a first session ready to run.

TEACHING NOTES

How to teach Day 10 — and the whole series

OPEN WITH THIS

Ask the room to name, fast, every Claude surface they have met — then ask the harder question: "What did all of them have in common?" The habits they surface are the real curriculum. Today makes that explicit and hands them the chalk.

PACE & EMPHASIS

This day is synthesis — resist re-teaching. Spend the time on the surface map (02), the six habits (03), and the handover to teaching (04). The capstone Lab is long and worth it; protect its time. End on the arc, not on a feature.

DISCUSSION PROMPTS

· Which surface will you reach for most, and which will you have to consciously remember exists? · Which of the six habits is your weakest, honestly? · Who are the first three people you would teach this to, and which day would hook each of them?

COMMON MISCONCEPTIONS TO PRE-EMPT

"The surfaces compete — I should pick one."
They are doorways to one capability. Real projects use several; the skill is choosing per task.

"I need to be an expert before I can teach this."
You need the map, the habits, and honesty. Learning openly alongside the room is credible, not weak.

"When the features change, the series is out of date."
The habits are durable; the specifics are a snapshot. Verify the facts, keep the framework.

IF YOU ONLY HAVE 30 MINUTES Teach the surface map (02) and the six habits (03) — that is the portable core of the entire series. Do Lab steps 1–3. The teaching handover can be a closing charge: "you have the map and the habits — now pass them on."

Whole-Series Cheat Sheet

The surface map	Apps = a person thinking · Claude Code = a codebase · API = your software · Connectors/MCP = reaching your systems · Design = making a thing.
Be specific	Vague in, vague out — from the everyday brief to the design direction.
Structure complexity	Hard work needs visible shape — tagged prompts, scoped tasks.
Reason on hard things	Plan-first, think-it-through; a fast answer to a hard question is risky.
Stay in the loop	Review, approve, judge — the more Claude can do, the more this matters.
Verify against reality	Ground answers, run tests, check output — confidence is not correctness.
Iterate	First output is a draft; the good result is steered, on a later round.
Teach it	Open with the gap, concept before mechanics, hands on every day, pre-empt the misconception.
Stay current	Features move; habits don’t. Verify specifics against official docs, keep the framework.

Check for understanding — cumulative

Five questions spanning the whole series. A learner who can answer these is ready to teach.

1. Give the one question that sorts most “which surface” decisions, and the five answers it leads to.
2. Name the six habits that ran through every day of the series.
3. Why are the surfaces best understood as doorways rather than competing products?
4. Name two teaching principles the Teaching Notes modelled, and why each works.
5. Why is the series built on habits rather than features — and what does that mean when something changes?

> **Answer notes —** 1) “Who or what is doing the work?” → a person thinking = the apps; a codebase = Claude Code; your software = the API; reaching your systems = connectors/MCP; making a thing = the design partnership. 2) Be specific; give complexity structure; ask for reasoning on hard things; stay in the loop; verify against reality; iterate. 3) They are one capability with different entry points — real projects use several at once, so the skill is matching the job to the doorway, not picking a side. 4) Any two — open with the gap (curiosity before content), concept before mechanics (the model carries the feature), hands on every day (people learn by doing), pre-empt the misconception (cheaper than correcting later). 5) Features and limits evolve but the habits are durable; when something changes you verify the specific against official docs and update the detail, while the framework still holds.

The series in five lines

- Claude is one capability with several doorways — apps, Claude Code, the API, connectors, the design partnership.
- Choose the surface by asking who or what is doing the work.
- Six habits run through all of it: be specific, structure complexity, reason on hard things, stay in the loop, verify, iterate.
- Teaching it well means opening with the gap, leading with the concept, and getting hands on every day.
- Features will change; the habits will not — which is exactly why this was worth learning, and worth teaching.

AFTER DAY 10 — THE BEGINNING

→ You have the map and the habits. Start your project, and teach someone else the way through.