A CRASH COURSE FOR EVERYONE · AGES 12 AND UP

Markdown in. HTML out.

How to give an AI instructions it can't misread — and get back pages people actually want to read. No coding needed.

ABOUT 40 MINUTES, SELF-PACED · WORKS WITH CLAUDE, CHATGPT, OR GEMINI

BEFORE WE START

The Problem: Two Leaks

YOU → THE AI

You write a messy paragraph.

So the AI has to guess what you meant. A wrong guess doesn't look wrong — it looks confident.

THE AI → YOU

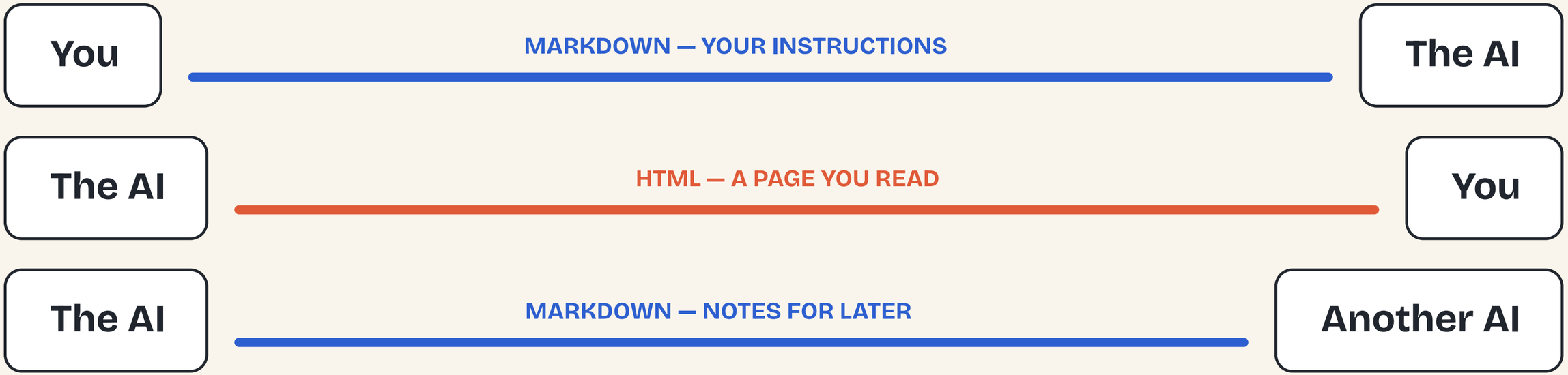
It answers with a wall of text.

Long, gray, all the same shape. So you skim it instead of reading it.

Both sides of the conversation leak. This course closes both leaks — with two small languages.

THE ONE PICTURE TO REMEMBER

The Big Idea: Two Languages



Markdown is plain text with a few symbols — #, -, ``` — that make the structure obvious. You'll learn it in Part 2.

HTML is the language web pages are made of. You never write it — the AI does. You learn to ask for it in Part 3.

WHAT'S AHEAD

The Course Map

PART 1

The Two Languages

Why neat text beats messy text, and which language goes in which direction.

PART 2

Markdown — the writing language

The five small things you write by hand, plus the master recipe: the spec.

PART 3

HTML — the reading language

What to ask for, how to ask well, and the one place the rule flips.

PART 4

Practice

One real exercise, three ways to run it, and how to share what you make.

Eleven small ideas in four parts. Each idea fits on one slide.

PART 1 OF 4

The Two Languages

Why neat text beats messy text — and which language goes in which direction.

PART 1 · IDEA 1

Structure Beats Guessing

The messy way

```
help me plan my birthday party, around 12 friends, we want pizza but Zara is vegetarian, budget is 6,000, mom says home by 9pm, oh and make a games list too
```

The AI must guess: is the budget strict? Which thing matters most?

The Markdown way

```
# Birthday Party Plan ## Hard rules - Budget: 6,000 total - Everyone home by 9:00 pm - Zara is vegetarian ## To plan - Pizza order that fits the budget - A list of 5 party games
```

Same facts. In the second one, the AI doesn't have to guess what matters.

PART 1 · IDEA 1

The One Rule to Remember

Anything the AI must not get wrong gets its own heading or bullet — never hidden inside a sentence.

✗ Buried mid-sentence: "...plan the sports day, the usual races, and keep the little kids out of the afternoon heat." The plan put the youngest race at 2 pm anyway.

✓ As its own bullet under "## Hard rules": "All under-8 races finish before 11:30 am." The next plan followed it perfectly.

The rule didn't change. Its visibility did.

PART 1 · IDEA 2

The Asymmetry

DIRECTION	LANGUAGE	WHY
You → the AI	**Markdown**	Structure removes guessing. Fast to type.
The AI → you	**HTML**	Rich, readable, and easy to share as a link.
The AI → another AI	**Markdown**	Compact and precise. Decoration is just noise to a machine.

You hand-write only the first row. The second row you never write — you ask for it.

PART 1 · IDEA 2

Who Reads It Last?

One question settles every format choice.

A person, in a browser	HTML
Another AI	MARKDOWN
A person, scrolling a social feed	PLAIN TEXT

The feed one is a special case — we'll come back to it at the end of Part 3.

PART 2 OF 4

Markdown: The Writing Language

Five small skills. About an hour to learn. You'll use them for years.

PART 2 · IDEA 3

Headings: The Skeleton

```
# Document title   ← one per document ## Big section ### Smaller section
```

Headings tell the AI which information rules which. Everything under "## Hard rules" is a rule — no guessing needed.

1. One # title per document. Two titles look like two tasks glued together.
2. Never skip levels (# straight to ###). A broken ladder confuses what belongs where.
3. Make headings claims, not labels. "## Budget: 6,000 max" beats "## Budget".

PART 2 · IDEA 4

Lists: Sets and Sequences

Bullets mean a set

```
- Pizza order - Games list - Music
```

Order doesn't matter. Each item stands alone. Use for rules, wishes, and features.

Numbers mean a sequence

```
1. Send the invites 2. Order the pizza 3. Set up the games
```

Order is the point. The AI won't start step 3 before step 1.

This isn't decoration. The list type tells the AI how to treat the items — and it takes the difference seriously.

PART 2 · IDEA 5

Code Fences: Quoting Exactly

Three backticks ``` around text mean: "this is exact data — don't act on it, don't reword it." The most useful thing to fence is an example of the answer you want:

```
The schedule table must look exactly like this: | Class      | Day | Time    | | ---------- | --- | ------- | |
Math help  | Mon | 4:00 pm | Same columns, same order.
```

One example row replaces three paragraphs of explaining — and the AI can't misread it.

PART 2 · IDEA 6

Links and Pictures

```
[the school rules](school.com/rules)
```

A link is context the AI can fetch.

Don't retell the rules from memory — and maybe get them wrong.

Link them and say: "match this."

```
![poster: title on top,
dates below](poster.png)
```

The description is what the AI reads.

When it can't see the picture, the words in brackets are all it gets.

Use them to say what to notice.

That's all five pieces: headings, lists, fences, links, pictures. Now let's assemble them.

PART 2 · IDEA 7

The Spec: Your Master Recipe

```
# What you're making ## Goal One short paragraph.
## Requirements - Things it must have ## Hard rules
- Things it must never break ## Out of scope -
Things you are NOT asking for ## Expected output An
example of what "done" looks like.
```

A spec (short for specification) is a recipe for the AI: everything you want, written down clearly. This skeleton works for anything — a party, a poster, a project.

Out of scope

Stops the AI from building extras that bury the three things you actually wanted.

Expected output

Stops it from drifting into its own format. Show one example of "done".

PART 2 · IDEA 7

Grade Your Spec

Don't hope your spec is good. Before the AI builds anything, ask it to attack the spec itself:

```
Here is my plan: [paste your spec]

Don't build anything yet. First:
1. List every spot two people could read differently.
2. List every rule I forgot to write down.
3. Grade it out of 10 for clarity and completeness.
4. Tell me the one change that raises the grade most.
```

COPY

Fix, regrade, repeat — until it scores 9 or more. A fuzzy bullet costs a minute to fix now, and an afternoon later.

PART 3 OF 4

HTML: The Reading Language

You never write it. You learn to ask for it — and to judge what comes back.

PART 3 · IDEA 8

Why Ask for HTML?

A long answer in plain text gets skimmed. The same answer as a designed page gets read. That difference is the whole argument.

WHAT HTML ADDS	WHAT IT LOOKS LIKE
Density	Tables, color, and diagrams — instead of walls of bullets.
Navigation	A contents list, tabs, sections that fold away.
Sharing	It's just a link. Anyone with a browser can open it — no app, no account.
Interaction	Sliders and buttons. The page becomes a tool, not a printout.

Honest caveat: for short answers, plain text is still right. Ask for HTML when the answer is long, visual, or shared.

PART 3 · IDEA 9

You Don't Write HTML — You Order It

The trigger is one sentence: "make this an HTML artifact" (Claude) or "put this in Canvas" (ChatGPT, Gemini). The quality comes from your brief — four things:

1. **The reader.** Who opens it, and on what? "My class, on phones."
2. **The parts.** Name them: one diagram, one table, a short tips list at the bottom.
3. **The interaction.** Sliders for what? A copy button that copies what?
4. **The reading mode.** Read once? Come back often? Show to an adult?

PART 3 · IDEA 10

The Five Useful Shapes

1 · Plans

"Show me 5 options side by side. Don't pick one yet — let me look first."

2 · Explainers

A pile of notes or data becomes a one-page picture with the key facts on top.

3 · Code review

Changes shown as visuals instead of raw text. For when you start coding.

4 · Prototypes

Sliders and knobs for things that are hard to say in words — like a color.

5 · Throwaway editors

Drag cards to sort your tasks, press "copy as text," then toss the page away.

Every shape, same rhythm: text in → HTML out → text comes back.

The page is the throwaway; the text it gives back is the thing you keep.

PART 3 · THE SPECIAL CASE

The Feed Exception

WhatsApp, Instagram, and LinkedIn never show your HTML. The feed pours your words into its own box.

YOU'RE MAKING...	USE
A WhatsApp message	Plain text, short lines.
A post	Plain text — make the first line count.
A page you link from the post	HTML — with preview tags, so the link card looks designed.
A picture for the post	HTML turned into an image. Ask the AI to render it as a PNG.

Pasting raw HTML into a chat is always wrong — it lands as a wall of angle brackets.

PART 4 OF 4

Practice

One exercise. Three ways to run it. Start with chat — today.

PART 4 · IDEA 11

Three Ways to Work

START HERE

Chat

Claude.ai · ChatGPT · Gemini

You paste your notes into the conversation. The main method — everything in this course runs here.

LATER

Terminal

Claude Code · OpenCode

The AI reads whole folders of files by itself. For bigger jobs, once chat feels easy.

LATER

Desktop

Cowork · OpenWork

Point it at a folder. It proposes a plan first, and touches nothing until you approve.

The skill is the same everywhere. Only the surroundings change.

PART 4 · IDEA 11

Your First Real Exercise

1. Grab real messy notes — a project, a trip, a study plan.
2. Ask for **spec.md** — your notes rewritten in the master recipe. Grade it to 9+.
3. Ask for **report.html** — a one-page version of the spec for a real person.
4. Open it on your phone. Would you proudly send the link? If not, say why — and regenerate.

Two files: precise **Markdown** for the machine, rich **HTML** for the human. That pair is the whole skill.

PART 4 · SHIPPING IT

Sharing Your Page

Making the page and publishing it are two different steps. Four rungs, easiest first:

YOU WANT…	USE
A link to share right now, zero setup	**Claude's Publish button**
To keep and edit the file yourself	**GitHub Gist**
A lasting page with your own address	**GitHub Pages**
Drag-and-drop hosting, no code at all	**Netlify Drop**

Pick the lowest rung that does the job.

PART 4 · TRY THESE TODAY

Prompts to Try Today

1 · The restructure

COPY

```
Here are my messy notes:
[paste them]

Rewrite them as a Markdown
document with headings for
Goal, Hard rules, Already
handled, and Open
questions. If a fact could
fit two headings, pick one
and tell me why.
```

2 · The conversion test

COPY

```
Take your longest recent
answer to me and turn it
into a single HTML page
made for reading: clear
sections, the key facts as
a table, one simple
diagram. Easy to read on a
phone.
```

3 · The sorter

COPY

```
Here are 12 things I need
to do: [list them]

Make an HTML page with
cards I can drag into Now /
Next / Later columns, plus
a button that copies my
final order as a Markdown
list.
```

Tap Copy, paste into Claude, ChatGPT, or Gemini, and fill in the brackets.

KEEP THIS ONE

The Cheat Sheet

The five Markdown pieces

```
# Heading        structure - item          a set
1. item          a sequence ``` text ```    exact
quote [text](link)    fetchable source
```

Who reads it last?

A person, in a browser	HTML
Another AI	MARKDOWN
A social feed	PLAIN TEXT

The master recipe

Goal · Context · Requirements · Hard rules · Out of scope · Expected output

THE WHOLE COURSE IN ONE SENTENCE

Write Markdown precise enough for a machine. Ask for HTML rich enough for a human.

That's it. Now go run the three prompts.