// FREE GUIDE · 12 SECTIONS · BY JOSH POCOCK

Ink Diorama: the free guide

Make Ricks Clips style story Shorts with Claude, built in code

The free skill and every prompt from my video, in order. Give Claude one strange true fact and it writes the script, builds a 3D toon world in code and renders a vertical Short on your own computer, with no AI video generator.

TERMINAL · BOOT

```
# install once, then make a Short from one fact
> Install this skill, then run its doctor with --fix and its selftest
✓ READY
✓ SELFTEST PASSED
> Make me an ink diorama story Short about this fact: <your fact>
```

Get the skill on GitHub →

Read it on the web: strideaiacademy.com/r/ink-diorama

Claude never posts or uploads anything. You get an mp4 and a check report, and you decide what happens next.



SECTION 01

The one prompt

Paste this into Claude Code. It installs the skill, checks your computer, installs anything missing, runs a test render and then asks you for a fact.

PROMPT

```
https://github.com/joshpocock/ink-diorama
Install this skill, then run its doctor with --fix and its selftest, then ask me for a strange true fact.
```

01
Paste the prompt above
Into Claude Code, on Windows or Mac

→

02
Claude installs and tests
doctor --fix, then the selftest render

→

03
Give it one strange true fact
A person doing something that sounds stupid

→

04
Say yes to the first sentence
Nothing is built before your yes

→

05
Get your mp4
Plus the checks and a contact sheet

Then make a Short

Start a new Claude Code session so it sees the skill, then type this with your own fact:

PROMPT

```
Make me an ink diorama story Short about this fact: <your fact>
```

The one from my video was: `Make me an ink diorama story Short about this fact: Hanoi, 1902. The city paid one cent for every rat tail.`

By hand on Windows (PowerShell)

>_ TERMINAL

```
git clone https://github.com/joshpocock/ink-diorama.git "$HOME\.claude\skills\ink-diorama"; if ($?) { python "$HOME\.claude\skills\ink-diorama\scripts\doctor.py" --fix }
```

By hand on Mac (untested so far)

>_ TERMINAL

```
git clone https://github.com/joshpocock/ink-diorama.git ~/.claude/skills/ink-diorama && python3
~/.claude/skills/ink-diorama/scripts/doctor.py --fix
```

I built and tested this on Windows 11. The Mac commands should work the same way, but I have not run them on a Mac yet. If something breaks there, paste the error into Claude Code.

What you need: Windows 11 or macOS, Python 3.8 or newer, Node.js 22 LTS, ffmpeg, git, internet during renders and about 2 GB free. `doctor.py --fix` installs what is missing and asks before anything that needs your password. A graphics card helps a lot.

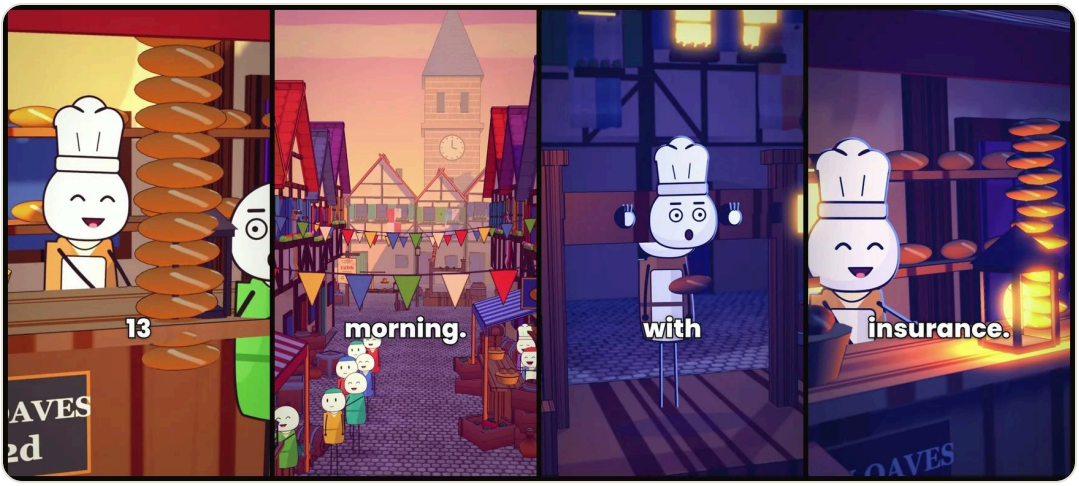
On the Claude app (claude.ai, no code running on your computer), the skill can write the script and the scene file, but the render needs Claude Code on your own machine.

SECTION 02

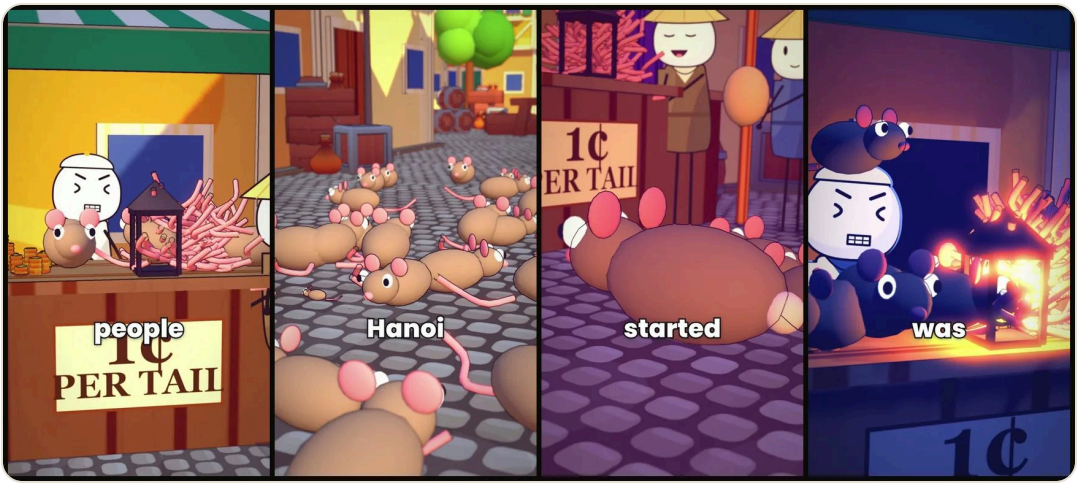
What you will make

A 20 to 30 second vertical story Short. Flat cartoon characters stand in a real 3D toon world with ink lines on its edges. The camera glides instead of cutting, the light goes from warm day to deep night blue, one white word sits on screen at a time and a fast narrator never stops. Every frame is built in code. No AI images, no AI video.

These are the two Shorts from my video, both made with this skill.



The baker's dozen, 22 seconds. Day, golden, then night blue from "You see" to the last word.



The rat bounty, about 26 seconds. Same town as the baker Short, repainted, with rats built in code.

Both previews are in the skill's `examples/` folder: `rat-bounty-preview.mp4` and `baker-preview.mp4`, made with the free voice.

SECTION 03

What Claude does at each step

STEP	WHAT HAPPENS
First run	Claude runs `doctor.py` until it says READY, then `selftest.py`, and opens the selftest contact sheet before it says anything works.
Fact	Checked in two sources. Claude tells you what is solid and what is retelling. If it is false, it says so and suggests a true one nearby.
Script	Six steps, 60 to 110 words. `check_script.py` fails question hooks, lesson openers and endings with a call to action.
Your yes	You see the first sentence on its own, then the full script. Nothing is built until you say yes.
Scene	One `scene.js`: a ready town or reading room re-skinned, a cast of card characters, camera moves and a light arc, each tied to a spoken word.
Voice	ElevenLabs if your key is set, otherwise Kokoro (free, local). Every pause is cut and the take is paced to about 3.7 words a second. Or record your own.
Render	HyperFrames draws each frame with three.js: ink lines, painted textures, depth of field, bloom and the grade.
Sound	Whooshes, hits and ticks plus a soft music bed, all made in code, mixed to -14 LUFS.
Check	Size, length, pace, loudness, blank frames, and a contact sheet Claude must look at. Then it fixes the worst three things, at most three rounds.

At the end Claude hands you the video path, the script, the checks, the render time, the voice it used and an honest list of what is still weak. Then it asks if you want any moment changed.

SECTION 04

The first sentence decides the views

This is the part every tutorial skips. On Ricks Clips, the same style in the same month got 34 million views on one Short and 135,000 on another. The cartoon was the same. The first sentence was not.

Every winner opens the same way: a person does something that sounds stupid or impossible, stated as a fact, with no setup. "They buried a man alive with a bell tied to his finger" got 12 million. The flops open like a lesson: "How Painkillers Actually Work" got 319K, "The Science Behind Slot Machines" got 135K.

The rule: if your first sentence is a question, you already lost.

The six-step formula

#	STEP	THE BAKER'S DOZEN
1	**A paradox, stated as fact.** 10 to 16 words. No question, no "How" or "Why".	"A baker gave every customer thirteen loaves, but only charged them for twelve."
2	**The setting**, in one line.	Folded into step 3 here.
3	**People laugh at him.** It keeps happening.	"The other bakers laughed. You're giving bread away! He did it every single morning."
4	**"But why?"** The viewer's own question, said out loud.	"But why?"
5	**"You see,"** and the real reason. The light turns night blue here.	"You see, in twelve sixty-six, the king made selling bread that was too light a crime..."
6	**An ironic last line** that flips the first. The video ends on its last word.	"So that thirteenth loaf wasn't kindness. It was insurance."

60 to 110 words in all, which is 18 to 30 seconds at the format's pace. Write numbers as spoken words ("nineteen oh two") so the voice and the captions match. My baker script is 81 words and 22 seconds, and it is a true story. Check yours before you build it, because the comments will.

Fill-in prompts

No fact yet? Ask for some:

PROMPT

```
Use the ink-diorama skill. Give me five strange true facts that would make a good story Short. Each one needs a person or a city doing something that sounds stupid or impossible, a hidden reason that makes sense, and two sources. Tell me which part of each is solid and which part is popular retelling.
```

Have a fact? Fill in the six steps yourself, or leave the slots and let Claude fill them:

PROMPT

```
Use the ink-diorama skill. Write my script in the six-step formula, 60 to 110 words, numbers as words:
1. Paradox as fact: <who> <did what, that sounds stupid or impossible>
2. Setting: <where and when, one line>
3. Doubt: <who laughed, and what kept happening>
4. But why?
5. You see, <the real reason>
6. Ironic last line: <the line that flips the first one>
End on the last word. Run check_script.py, fix every FAIL, then show me the first sentence on its own.
```

Don't love the first sentence? Say no and ask for three more:

PROMPT

```
Give me three new first sentences for this fact, each from a different angle: the person, the number, the result. Each one a statement of 10 to 16 words. No question and no lesson opener.
```

Wrote your own script? Check it before you build:

PROMPT

```
Save this as the script for a new ink-diorama job and run check_script.py on it. Tell me every FAIL and WARN and how to fix it:
<paste your script>
```

Good facts: a rule or reward that produced the opposite of what it meant to (the rat bounty), a strange custom with a sensible hidden reason (the baker's dozen), a date or number that sounds impossible. Weak facts: no person in it, needs a diagram, or the "why" is still argued about. Write your own stories. Don't remake another channel's.

SECTION 05

The five measured rules

The first sentence gets the click. These five keep people watching. They come from measuring 20 Shorts in this style frame by frame, and every step of the build is there to hit one of them. The skill is already set to all five.

1. **The voice never stops.** About 3.7 words a second, and less than one second of silence in the whole Short.
2. **Almost no cuts.** Zero or one hard cut in a whole Short. The camera is moving on about two thirds of all frames and glides from one moment to the next.
3. **One white word at a time.** Low in the frame, at about 69% of the height, swapped on the exact frame each word is spoken. No title, no logo.
4. **The light tells the story.** Warm day for the set-up, golden for the build-up, deep night blue the moment the narrator says "You see".
5. **It ends on the last word.** No outro, no "follow for more". So it loops, and the loop counts as another view.

SECTION 06

The build, step by step

Claude does all of this from the one prompt. This section is for when you want to change something. Your Short lives in `ink-shorts/<slug>/` in the folder where you ran Claude, and a new story is one new `scene.js`. Everything else is already in the skill.

In the commands below, `S` is the skill's scripts folder: `$HOME\.claude\skills\ink-diorama\scripts` on Windows, `~/.claude/skills/ink-diorama/scripts` on Mac. After any scene change, render again without remaking the voice:

```
>_ TERMINAL

python S/make.py all ink-shorts/<slug> --reuse-voice
```

1. The voice

The voice goes first, because everything else is timed to it. My first take came out at 2.8 words a second, so the skill cut every gap and sped it up to 3.6. Then it finds the exact frame of every word, and every camera move and face change lands on a word. Free voices: am_michael, am_adam, bm_george, af_heart, af_nova, bf_emma.

```
PROMPT

Remake the voice for ink-shorts/<slug> with the free Kokoro voice <bm_george>, then render a new
version.
```

```
PROMPT

Use my own recording <take.m4a> as the voice for ink-shorts/<slug>. Cut the pauses, pace it, and
render a new version.
```

For ElevenLabs, set your key in the terminal before you start Claude (it stays in your environment, never in a file). The skill uses it automatically:

```
>_ TERMINAL

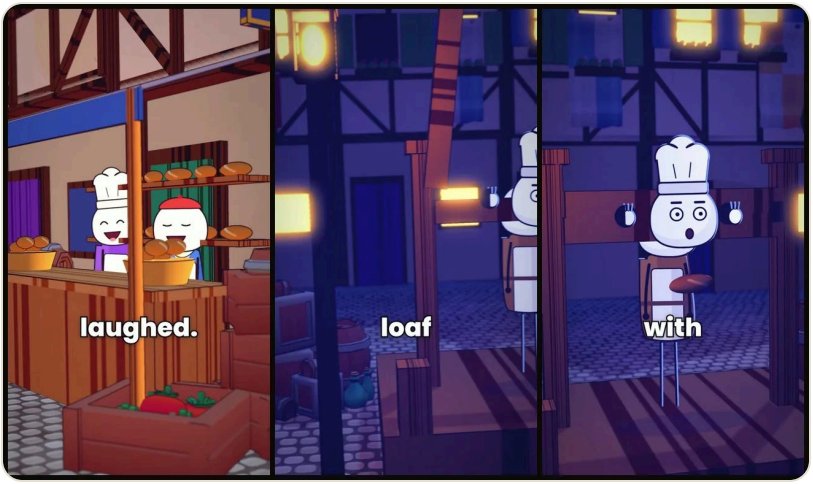
$env:ELEVENLABS_API_KEY = "<your key>"   # Windows PowerShell, this window only
export ELEVENLABS_API_KEY="<your key>"   # Mac
```

2. The world

Real 3D, written as code. Every house is boxes, beams and a roof with a painted texture Claude draws itself. Barrels, crates and coins are free CC0 3D models included in the skill. A thin dark ink line on the edges makes it look hand-made. You get a market town (street, two stalls, a square with a pillory and a tower) and a reading room. Re-skin them instead of building new worlds. That is how the rat Short reused the baker's town.

PROMPT

```
In ink-shorts/<slug>/scene.js, re-skin the town for <place and year>: <wall colours>, <shutter colours>, add <props>. Keep the same street and square, then render with --reuse-voice.
```



The same town in day and night. The light switch to night happens inside a camera move, never on a held shot.

3. The characters

Flat 2D drawings standing in the 3D world, like paper cut-outs, drawn in code: a round white head, a simple body, a hat so you know who is who. They take the colour of the scene, warm by day and blue at night, so they never look pasted on. Each one has poses (stand, walk, point, laugh, run, carry, pillory and more) and faces (neutral, happy, sly, shock, laugh).

PROMPT

```
In ink-shorts/<slug>/scene.js, add <a character in a red coat>. On "<word>" he <points at the scale>, and his face goes to <shock> on "<word>". Land each face two frames before its word. Keep a face in every shot.
```

4. The camera

The rule most people break. These Shorts almost never cut. The camera glides, keeps pushing in slowly on quiet lines, and changes place by travelling or whipping, never by cutting. A cut is a place to scroll away. My baker Short has one hard cut in 22 seconds.

PROMPT

```
In ink-shorts/<slug>/scene.js, open with a slow push in on <the hero>, whip to <the square> on "You see" and switch to night inside that whip, and add a short shake on "<impact word>". No more than one hard cut in the whole Short.
```

5. The sound

The one people forget. The voice is mixed to -14 LUFS, the level YouTube plays everything at. The music bed sits about 8 dB under it. Effects go only on travel and impacts, and every sound is made in code: whoosh, whoosh-short, swipe, tick, thud, hit, snip, clank, pop, chime. You don't set any of this by hand.

PROMPT

```
In ink-shorts/<slug>/scene.js, add a <thud> on "<word>" and a <whoosh> on the whip to <place>. Effects on travel and impacts only. Keep the mix at -14 LUFS and render with --reuse-voice.
```

No music bed at all: add `--no-bed` to the make command.

SECTION 07

One Short, start to finish

To rebuild the rat bounty Short exactly as it ships, two commands:

```
>_ TERMINAL

python S/make.py new rats --from rat-bounty
python S/make.py all ink-shorts/rats
```

Real output on a Windows 11 PC with an RTX graphics card, installed into a fresh folder, free Kokoro voice:

```
>_ OUTPUT

voice: ink-shorts\rats\audio\voice.wav (26.31 s, 3.65 words/s, kokoro)
words: 96 timed by transcribed (94 heard, 93 matched the script)
render: ink-shorts\rats\renders\v1.mp4  (picture 136 s, total 147 s, gpu: hardware gpu)
PASS  size      1080x1920 at 30.0 fps, 794 frames, 26.47 s
PASS  length    ends 0.19 s after the last word
PASS  pace      3.65 words a second (format: about 3.7)
PASS  loudness  -14.1 LUFS, true peak -2.2 dBTP
PASS  blank     no isolated blank or flash frames
PASS  dark      no near-black frames
```

The picture took 136 seconds while other renders were sharing the graphics card, and 65 seconds when the card was free. The whole first run (voice, timings, render, sound, checks) took 341 seconds.

SECTION 08

Get it reviewed blind

My first version looked fine to me. So I had a second Claude review it blind. It only saw frames from the real channel and from my Short, and I told it to be harsh. It scored my first version 4 out of 10 on "survives the feed". Don't post your first render.

1. Take a few screenshots of the Shorts you are aiming for and put them in a folder. They are for your own review only. Never put them in your video.
2. Open a **new** Claude Code session, so the reviewer has no memory of building your Short.
3. Paste the prompt below with your two paths filled in.

PROMPT

```
You are a blind reviewer. You did not make this video and you cannot see its code or notes. The folder <reference folder> holds frames from the Shorts I am aiming for. The image <ink-shorts/slug/qa/vN/contact-sheet.jpg> is the contact sheet of my Short. Open every image at full size before you score anything. Be harsh: your job is to find what is wrong, not to reassure me.
Score my Short 1 to 10 on: style match, composition, light, ink, characters, camera, captions, story, and survives the feed (would someone stop scrolling, and would it pass as an upload from the channel I am aiming for). 8 means it would pass.
For every score under 8, name the frame and what to fix. End with the three worst problems, worst first.
```

Then go back to the session that built the Short and fix only those three, so the next review measures the fix:

PROMPT

```
A blind reviewer named these three problems in ink-shorts/<slug>:
1. <problem>
2. <problem>
3. <problem>
Fix only these three in scene.js and render a new version with --reuse-voice. Don't overwrite the old render.
```

Use a fresh reviewer every round. Each render takes about a minute, so a fix is cheap. My baker Short took 15 versions and 5 blind reviews.

The honest limits

My baker Short went from 4 to **5 out of 10** on "survives the feed", and the rat Short ended at 5 too. The bar is 8, and neither reached it. Know this before you start a channel with it.

BLIND SCORE, OUT OF 10	BAKER	RAT BOUNTY
Style match	6	6
Composition	5	6
Light	6	7
Ink	6	5
Characters	6	6
Camera	7	6
Captions	6	6
Story	5	6
Survives the feed	**5**	**5**

- The real channel still fits more into every frame. My sets are thinner.
- The characters are stiff. Faces change, but nobody really walks yet.
- Night comes out more violet than deep blue.
- The code-built rats read as smooth clay, so keep animals small, many and moving.

Treat the skill as a strong start, not a finished channel. The blind review is how you close the gap.

SECTION 09

How these channels earn (an estimate)

I checked Ricks Clips: no sponsors, no links, no products. Its money is YouTube's ad share on Shorts, and nothing else.

vidIQ puts most channels at 3 to 10 cents for every 1,000 Shorts views. The channel switched to this style on October 7th and has had 29 animated Shorts and about 262 million views in five weeks since.

```
>_ OUTPUT

262,000,000 views / 1,000 = 262,000
262,000 x $0.03 = $7,860
262,000 x $0.10 = $26,200
ESTIMATE, AD SHARE ONLY
```

So roughly $8,000 to $26,000 from ad share alone. This is an estimate from a public rate range, not the channel's real earnings and not a promise of yours.

SECTION 10

Fixes

Most of these Claude handles for you. If you get stuck, paste the error into Claude Code.

- **Doctor says NOT READY.** Run `doctor.py --fix`, then open a **new** terminal so the new tools are on your PATH, and run doctor again. Repeat until it says READY.
- **Claude doesn't know the skill.** Start a new Claude Code session after installing.
- **The render is very slow, or the log says software.** Chrome drew the 3D without your graphics card. It works, but many times slower. Close other apps using the card (the same Short took 136 seconds with the card shared and 65 seconds with it free).
- **Cannot reach the three.js CDN.** Renders load three.js from cdn.jsdelivr.net, so you need internet while rendering.
- **The free voice did not run.** Run `doctor.py --fix` again; it downloads the Kokoro voice once. Until then the skill falls back to your computer's own voice, which sounds robotic. Or record your own take.
- **Word timings failed.** Add `--estimate` to the make command. It spreads the words over the speech by length, which is good enough for a draft.
- **ELEVENLABS_API_KEY is not set.** Set the key in the terminal before you start Claude, or use `--engine kokoro` for the free voice.
- **check_script.py says FAIL.** Your first sentence is a question or opens like a lesson, the script is outside 60 to 110 words, or it ends with a call to action. Rewrite before you build.
- **A check says FAIL after the render.** Pace means the voice is too slow or too fast (target 3.3 to 4.2 words a second). Length means the video runs more than half a second past the last word. Loudness means it is off -14 LUFS. Blank means a blank or flash frame. Ask Claude to fix it and render a new version.
- **The timing broke after you changed a line.** Every moment in a scene is tied to a spoken word, never to a typed frame number. Ask Claude to check that the CUES list is in spoken order.
- **Not enough space.** Renders need about 2 GB free.
- **Where did my old version go?** Nowhere. Renders are saved as v1, v2, v3 in `ink-shorts/<slug>/renders/`, and the skill never overwrites one.

SECTION 11

Before you start a channel

1. **Spend your time on the first sentence.** 34 million against 135,000, same cartoon.
2. **Write your own stories.** Don't remake theirs.
3. **Don't post the first render.** Get it reviewed blind, by Claude or by a friend, and fix what they say.

SECTION 12

The fine print

It never posts. Claude never posts, uploads or publishes anything. Everything stays on your computer in `./ink-shorts/`.

True stories, your own. Check every fact in two sources. Never retell a story another channel already made, never copy anyone's characters, footage, music or channel name, and never put a living private person in a Short.

Built in code. No AI image or video generation. Props are free CC0 models by Kay Lousberg (KayKit). The caption font is Poppins (OFL). Sounds and music are made in code. The full list is in `references/licences.md`.

HyperFrames is HeyGen's open-source renderer (Apache 2.0), run with npx. It is not included in the skill.

Tested on Windows 11 with Node 22 and an NVIDIA card. Untested on Mac so far.

The earnings section is an estimate from a public rate range and a public view count. It is not anyone's real earnings.

Not affiliated with Anthropic, HeyGen, ElevenLabs, vidIQ or Ricks Clips. Ricks Clips is named only as the format this style comes from.

Free and MIT licensed. No signup, and no API key needed.

Get the next free skill

Stride AI Academy is my free community, with over 1,200 people. It's where I give away the Claude skills I build, with a written guide like this one. If you want more, Stride AI Academy Pro adds StrideOps, weekly calls with me, and the skills and content I don't post anywhere else.

Join Stride AI Academy free →

THE ACADEMY	strideaiacademy.com Courses, weekly live calls, and the build-it-with-me community.
THE SKILL	github.com/joshpocock/ink-diorama MIT licensed. Star it if it saved you.
THIS GUIDE, LIVE	strideaiacademy.com/r/ink-diorama Copy buttons on every prompt, always the latest version.
EVERY FREE SKILL	Email me every free skill Optional. The library of free skills and guides, by email.

STRIDE