scenario

SEVEN WIND-UP TIN TOYS / MAKING OF

Wind-Up Toys

Seven mechanical tin toys, each built from one concept image and one written brief by the new Claude Fable 5.2 3D model on Scenario. One file comes back with the mesh, the painted textures, the skeleton and the animations.



7	1	50 min	2,100
wind-up toys	prompt per toy	for one build	CU for one build

One picture per toy.

Each toy started as an idea in a few words, then one concept image. All seven share one painted tin-toy look, so they read as one family.



Robo, the crab.



Tangle, the octopus.



Sun Beetle: picked from four insects.



Pangolin: picked from four armadillos.

The concept image

Made with GPT Image 2.5 Sunburst on Scenario, quality high, 12 to 13 CU a picture. Two earlier toys, Robo and Tangle, went in as style references so the new ones match. The style words asked for "many visible moving parts: gears, pistons, hinges, linkages, springs".

Why a picture first

The 3D model takes the shape, the colours and the paint from the picture. The written brief then adds what a picture cannot show: which parts move, where each one turns, and which animations to make.

One connection to every model.

The agent in Claude Code reaches Scenario through one connection, the Scenario MCP. Through it the agent picks a model, reads its settings, starts a job and collects the result.



The idea goes to Claude Code. The Scenario MCP links it to the image model and to Claude Fable 5.2 3D.

What the agent did

It turned the idea into a concept image, sent that image with a written brief to the 3D model, waited for the build, downloaded the file and checked it. No other tool sat between the agent and the models.

A long job

One Claude Fable 5.2 3D build takes about 50 minutes; Robo took 49. The agent starts the job and checks back on it, rather than waiting on a single call.

The brief names every part.

One request per toy: the concept image and a brief of under 2,000 characters. Model: Claude Fable 5.2 3D, build effort maximum, face budget 50,000, output GLB.



The only reference: Robo's concept image.



49 minutes later: Robo, as the file came back.



With its skeleton already inside.

Parts and rig, from the brief

PARTS

Every moving part is its own clean watertight named object with its pivot on its joint, nothing fused.

RIG

RIG: one armature, one bone per moving part placed at its pivot (root, body, turret ring, horn, key, each eye spring and eye, … each leg hip/upper/knee/lower/foot), parts rigidly bound to their bone, clear names like Leg_FL_Upper.

Animation, from the brief

ANIMATION

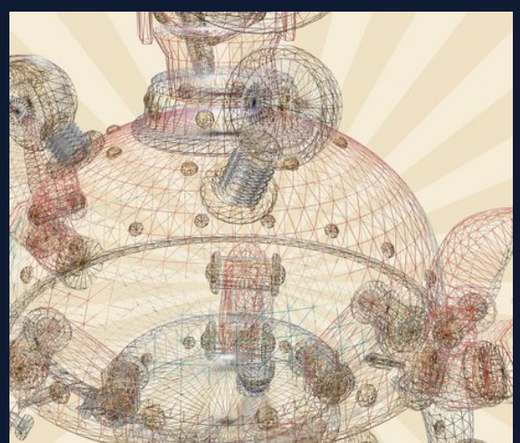
ANIMATION: export these looping clips as separate named actions in the GLB: Idle (…), Walk (diagonal-pair spider gait, feet planted without sliding, body bob), ClawSnap (…), TurretSweep (…).

TIP

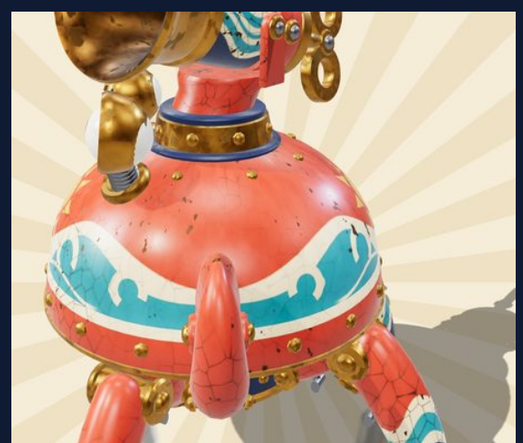
Ask for the rig and the clips in the same brief as the model: name every moving part, one bone per part, and the clips by name.

Mesh, paint, skeleton, clips.

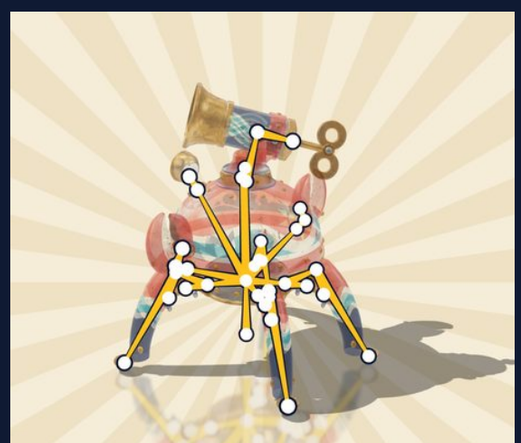
One GLB comes back with everything inside. There is no separate texturing step and no separate rigging step: Claude Fable 5.2 3D does it all from the prompt, inside Scenario.



The mesh: clean, separate, named parts.



The textures: painted by the model.



The skeleton: 43 bones, one for each moving part.



The animations: Idle, Walk, Claw Snap, Turret Sweep.

Rigged by the model

The skeleton arrives inside the file, with one bone at the pivot of every moving part and a clear name on each. Robo has 43 bones, the Pangolin 39 and the Sun Beetle 50.

Clips inside the file

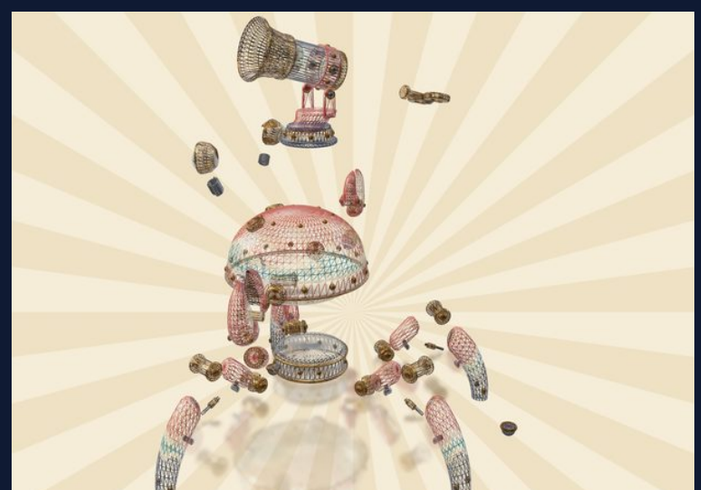
Each toy carries three or four looping clips, named in the brief: Walk, Hop, Roll Up, Hover and more. The page plays them straight from the file.

Every part is its own mesh.

Because each moving part is a separate object, a model comes apart cleanly. And the file opens in any 3D program, ready to change.



Robo, taken apart on the page.



The same parts in wireframe.



The Pangolin: every scale plate is its own part.

Checked before trusted

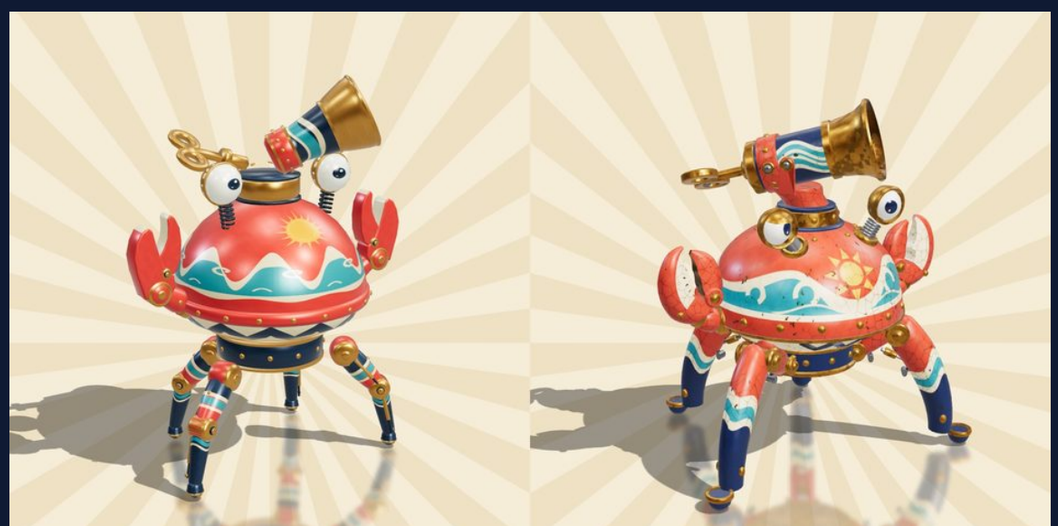
Every build was looked at before it went on the page: four views, a wireframe, and one pose from the middle of each clip.

One fix, in Blender

On the Sun Beetle, repeated parts such as legs, eyes and wings each came back tied a quarter to four different bones, so they drifted apart in motion. One Blender script bound every part to its own bone. No new generation was needed.

The same brief, built twice.

The same picture and the same words also went to a second 3D model on Scenario, Kimi K3.1 Pro 3D. On the page the two results turn side by side and play the same clips.



Robo: Kimi K3.1 Pro 3D on the left, Claude Fable 5.2 3D on the right.



The Pangolin: Kimi K3.1 Pro 3D on the left, Claude Fable 5.2 3D on the right.

Kimi K3.1 Pro 3D

Maximum effort, 50,000 faces. A build takes 10 to 30 minutes and costs 750 CU. Robo came back with 35 bones in a 6.0 MB file.

Claude Fable 5.2 3D

Build effort maximum, 50,000 faces. A build takes about 50 minutes and costs 2,100 CU. Robo came back with 43 bones in a 2.1 MB file.

Straight into a game.

The models can go into a game as they are. In the small browser game on the page, the Pangolin and the Sun Beetle are these exact Claude Fable builds, clips included.



Robo on the beach.



The Sun Beetle, landing.



The beach and its props.

The rest of the game

The game's ten props and its skybox were also made on Scenario: the props from a GPT Image 2.5 concept and Tripo P2, the sky as a Scenario Skybox panorama. The game itself runs on three.js in the browser.

The receipt

One concept image, GPT Image 2.5 Sunburst	12 to 13 CU
One build, Claude Fable 5.2 3D	2,100 CU
One build, Kimi K3.1 Pro 3D	750 CU

The cost of one generation each, from the session's Scenario job records.