AN INDEPENDENT EXPERIMENT · NOVEMBER 2026

February's AI made flashlight tag with invisible flashlights. November's made the game my kids play.

Same 679-word request, one try each, nobody helping. The AI from February stopped after 32 minutes. The one from November, from the same company, kept working on its own for nine hours. You can play both.

By Jon Bittner · November 18, 2026 · 6 minute read

Read it online and play the five games: dimensional-analytics.com/lab/fox-flashlight-tag





The same moment in both games: the fox has just rung a doorbell. No person edited either game.

It started with flashlight tag

One night my kids were out in the neighborhood playing flashlight tag. It gave me an idea for a fun little video game: a fox sneaking between the beams, getting into mischief, home before sunrise.

I work with AI every day, and I wanted a way to show people how much it has improved. A game seemed right. Anyone can tell a good game from a broken one in about a minute.

The request: 679 words, one try

So I wrote the game down in plain English. It came to 679 words, about a page and a half. You are a fox. The neighborhood kids are out with flashlights. Ring a doorbell and run. Be home in your den by dawn, five nights in a row. I listed ten things the game must have and left the rest to the AI, including the look, the characters and the jokes.

Then I gave those same words to five AIs. None of them is ChatGPT. All five are versions of Claude, made by a company called Anthropic: two released in February of this year, two in September and one in November. I ran all five this month on the same computer.

Each AI got one try, and it was not a chat. Think of the difference between asking someone a question and handing them a job. Each AI was given a computer and left to write the code, run it, look at the result and keep going until it decided the game was done. Nobody reviewed a draft, fixed a bug or suggested an idea.

Two things were not equal, and you should know them before you play. The three newer AIs ran in a mode that lets an AI work longer and split the job among helper copies of itself; the two February AIs did not. And better was not cheaper. At Anthropic's published prices the first February game cost about $6 to make and the November game about $593.

February: the flashlights never light up

Claude Sonnet 4.6 and Claude Opus 4.6 · released February 2026 · worked 32 and 57 minutes

The first game came back in 32 minutes. Start it and there it is: a small orange fox in a dark yard, with houses, hedges and a street. Half an hour earlier it was a page and a half of English.





Then you play. The kids are out, as asked, but their flashlights never light up. You creep past one and the screen flashes red.

You have been tagged by a beam you cannot see, in a game called Flashlight Tag. Most of the kids have walked into a garage or a hedge and stayed there. Three tags end the night, and the next night starts anyway. You cannot lose.

The second February AI took 57 minutes. Its flashlights work, but they shine straight through houses, and every kid is stuck against a wall or a tree within about 13 seconds.

The writing is better than the game. In one of them a neighbor sets out garden gnomes as “fox deterrents.” The story adds: “The fox calls them trophies.” Neither is a game you would play twice.

September: the street gets a name

Claude Sonnet 5.5 and Claude Opus 5.5 · released September 2026 · worked 4 hours 21 minutes and 3 hours 23 minutes

The next two AIs are the September versions of the same two products, and they did not come back in half an hour. One worked for more than four hours, the other for more than three.





This time there is a neighborhood. In one game the street is Maple Lane and the kids have names. Their flashlight beams stop at fences and parked cars. Ring a doorbell and a porch light snaps on. Get clear, and the door opens on a neighbor who mutters, “Hmm. Nobody.” On the fourth night an ice cream truck pulls every kid out of the game and into a line.

In the other game the kids meet every morning to go over what you did. “2 of Grandma’s gnomes went MISSING. She called the police. Well. She called my uncle.”

Both games do everything I asked for, and they share one flaw. Your den is safe and nothing makes you leave it, so you can win by never going out.

November: nine hours on its own

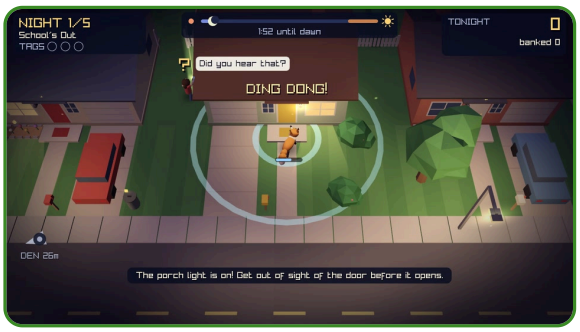
Claude Fable 5.2 · released November 9, 2026 · worked about 9 hours

The newest AI worked for about nine hours, longer than a school day.

Its game has eight kinds of mischief, six kids with their own habits and a beagle called Biscuit. Its opening story stops at the rules of flashlight tag, chalked on the sidewalk. “Nowhere do they say NO FOXES,” it says. “It checked.”

It also has a joke that runs the length of the game. Every morning the kids argue at their lemonade stand about what is behind all the mischief, and every morning the sign on the stand has changed. It starts at LEMONADE 50¢. By the second morning it reads RACCOON FACTS 10¢. One kid, Priya, says “It’s a fox” from the very first day. Nobody listens to Priya. I will not spoil where the sign ends up.

This is the one my kids play.



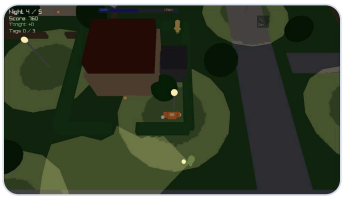
What it did for nine hours

The February AIs built the game, opened it and looked at pictures of the screen, 12 for one and 57 for the other. Then they stopped. A picture shows that a game starts. It cannot show that a kid has been walking into a garage for two minutes.

The invisible flashlights are the clearest case. The code for the beams is all there, and it even stops them at hedges. One small mistake tells the computer that every beam is facing the wrong way, so it never draws them. Switch that one rule off and they appear, as the second picture shows.



As built: the fox hides behind a hedge as a kid walks past. There is no beam to hide from.



The same moment with one drawing rule switched off as a test. The beam was there all along.

The November AI kept going. It rebuilt and re-ran the game about a hundred times. Then it wrote robot players to play it. I had asked for a game a beginner could start and only a skilled player could finish. It could not test that on people, so it tested it on its robots: a clumsy one almost always gets through the first night but finishes all five nights fewer than one time in ten, and a careful one finishes more than nine times in ten. Then it added a warning of its own: “People are not bots, so treat those as a guide.”

By the end it had written more code to check the game than to make it.



What it still gets wrong

The November game is not perfect. Score numbers sometimes land on top of each other. And its title reads FOH FLASHLIGHT TAG, because the X in its lettering looks like an H. It is the first thing on the screen, and it survived nine hours of checking. The AI looked at hundreds of pictures of its own game and measured everything it could. None of that was a person reading the title.

That is the pattern in the newer games. Those AIs test what they can measure: whether every yard can be reached, how hard the nights are, whether anything crashes. None of them can hear, so two of them checked their sound by measuring it. What is left over is where a person still has to look.

This test is one request and one try for each AI, so it is a demonstration and not a study. A second try would give a different game. How each game did against my ten must-haves is at the end: the February games manage eight and nine of them, and the other three manage all ten.

The plow did not retire the farmer

If an AI can do this much on its own, what is left for people? Most Americans expect the answer to be bad: in a Gallup survey this May, 79% said they expect AI to mean fewer jobs. One fox game does not settle that. Here is what I saw.

Building the game no longer needed a person. Three things still did: having the idea, writing it down clearly, and judging what came back. The November AI ran hundreds of test nights and still shipped a title that reads FOH. A person notices that in a second.

There is a line I like: the plow did not retire the farmer. The honest version comes with numbers. In 1900 about four in ten American workers were on farms. By 2000 it was fewer than two in a hundred, and most of the farmhands were gone. The farmer who remained had a different job, closer to a scientist with street smarts, working with soil chemistry, weather and prices.

I think that is the direction for anyone whose work happens at a keyboard, engineers and scientists included. Producing a first working version is getting cheap. Knowing what is worth building, checking it and answering for it are becoming the job. I think that will raise the level most of us work at. The farm numbers are a reminder that it will not be painless.

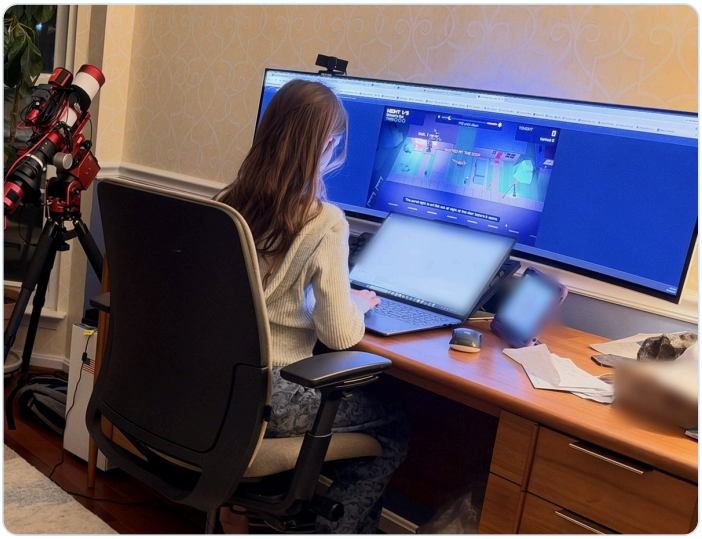
If you want the theory

Rich Sutton, The Bitter Lesson (2019): why simple methods with more computing power keep beating hand-built expertise.

Gwern, The Scaling Hypothesis (2020): the case that bigger systems keep getting more capable.

Casey Handmer, Where does Baumol bite? (2026): what stays scarce when thinking gets cheap. He names "Human attention, trust, and the willingness to delegate decisions."

Back to the street



One of my kids playing the November game.

"I'm obsessed with it," one of my kids told me. The game is about neighborhood kids out with flashlights, and my kids already have the neighborhood kids playing it.

In nine months, AI went from flashlights you cannot see to a game my kids chose to play. None of the five came any closer to seeing a game in kids playing flashlight tag. That part was mine.

The 679 words are below, ready for the next AI.

Play all five

Oldest first.



Claude Sonnet 4.6

FEBRUARY 2026 · WORKED 32 MIN

The kids are out, as asked, but their flashlights never light up.



Claude Opus 4.6

FEBRUARY 2026 · WORKED 57 MIN

Its flashlights work, but they shine straight through houses, and every kid is stuck against a wall or a tree within about 13 seconds.



Claude Sonnet 5.5

SEPTEMBER 2026 · WORKED 4 H 21 MIN

On the fourth night an ice cream truck pulls every kid out of the game and into a line.



Claude Opus 5.5

SEPTEMBER 2026 · WORKED 3 H 23 MIN

“2 of Grandma’s gnomes went MISSING. She called the police. Well. She called my uncle.”



Claude Fable 5.2

NOVEMBER 2026 · WORKED 9 H 2 MIN

Its game has eight kinds of mischief, six kids with their own habits and a beagle called Biscuit.

Try it yourself

If you last tried AI a year ago, try it again on something real. Write down what you want, what matters most and how you will know it worked. Then check the result yourself. The request I used is below; copy it and change the game to your own idea.

The details

What the prompt asked for	Sonnet 4.6	Opus 4.6	Sonnet 5.5	Opus 5.5	Fable 5.2
1. A 3D world seen from above, with a camera that follows the fox	✓	✓	✓	✓	✓
2. A neighborhood with houses, fences, hedges, trees and streetlights	✓	✓	✓	✓	✓
3. Kids with visible flashlight beams that tag the fox and are blocked by solid things Sonnet 4.6 never draws the beams. Opus 4.6 draws them through walls, and its kids stop moving.	Partly	Partly	✓	✓	✓
4. Ding dong ditch plus at least three more kinds of mischief	✓	✓	✓	✓	✓
5. Noise that brings nearby kids to look	✓	✓	✓	✓	✓

What the prompt asked for	Sonnet 4.6	Opus 4.6	Sonnet 5.5	Opus 5.5	Fable 5.2
6. Five two-minute nights, home by dawn to keep the points, three tags end the run In the Sonnet 4.6 game three tags end only that night, and points are kept without going home.	Partly	✓	✓	✓	✓
7. Every night adds or changes something and is harder	✓	✓	✓	✓	✓
8. An opening, a scene between nights, and an ending that reflects the run	✓	✓	✓	✓	✓
9. A title screen with the controls, an ending with the score, play again without reloading	✓	✓	✓	✓	✓
10. Keyboard only	✓	✓	✓	✓	✓
Working as asked	**8 of 10**	**9 of 10**	**10 of 10**	**10 of 10**	**10 of 10**

The checklist was filled in by AI play-testers (Claude) that played each game through with real key presses, kept screenshots and read the source code. It has not been scored by a panel of people, and it measures whether each feature works, not how good the game is.

Model	Released	Working time	Helper agents	Game code	Test tooling kept	Output tokens	List-price cost
Claude Sonnet 4.6	February 17, 2026	32 min	None	2,213 lines	None	112 thousand	$6.04
Claude Opus 4.6	February 5, 2026	57 min	None	1,856 lines	None	139 thousand	$13.50
Claude Sonnet 5.5	September 28, 2026	4 h 21 min	43	7,841 lines	72 lines	3.4 million	$125.39
Claude Opus 5.5	September 22, 2026	3 h 23 min	19	7,065 lines	1,145 lines	1.8 million	$111.01
Claude Fable 5.2	**November 9, 2026**	**9 h 2 min**	**35**	**7,715 lines**	**about 9,700 lines**	**5.0 million**	**about $593**

A token is roughly three quarters of a word; output includes the model's working notes as well as code. Costs are what the same work would cost at API list prices. The runs themselves were made on a subscription.

How it was run

- Every model received the same prompt, shown in full in this article, in a folder of its own.
- Same machine and tools for all five: a 16-core Linux workstation, Emscripten 3.1.74, raylib 5.5, and headless Chromium with Playwright for the models to check their work.
- Each ran in the Claude desktop app's coding mode with permission to run commands without asking. The two 4.6 models ran at high effort and worked alone. The three newer models ran at the highest effort setting with Ultracode on, which lets a model hand parts of a job to helper agents; they used 43, 19 and 35.
- The games are published unchanged, and their checksums are listed here. The frame around each game, the touch controls and the test that reveals the hidden beams are mine.
- Release dates are from press coverage of each launch. Working time counts the periods when a model or its helpers were active. The counts of builds, browser runs and screenshots come from each run's own log and are approximate.
- An independent experiment. Claude is made by Anthropic. The games are built with raylib and compiled with Emscripten.

Sources

- Farm employment figures are from the USDA Economic Research Service (41 percent of the workforce in 1900, 1.9 percent in 2000).
- The jobs figure is from a Bentley University and Gallup survey of U.S. adults in May 2026.
- Clips were recorded from the unchanged games at a fixed frame rate.

About the author

Jon Bittner builds AI systems at Dimensional Analytics. Anthropic, which makes Claude, had no part in this test.

Appendix: the request in full

```
Build a small 3D game that runs in the browser as WebAssembly. This is a single request: I won't be around to answer questions, so make your own decisions and keep working until the game is finished and you have checked that it runs.

THE GAME: FOX FLASHLIGHT TAG

It's summer in a quiet suburban neighborhood, and every night the kids are out playing flashlight tag. The player is a fox who has decided to join in uninvited. Night after night, the fox slips through backyards and across streets, stays out of the flashlight beams, and gets up to mischief along the way.

What the game must have:

1. View. A 3D world seen from above, with a camera that follows the fox.
2. Neighborhood. Houses, fences, hedges, trees, streetlights, and whatever else makes it feel like a real neighborhood at night. It should be fun to explore.
3. Flashlight tag. Several kids roam the neighborhood with flashlights. Each beam is a visible cone of light. A fox caught in a beam is tagged. Solid things block beams: a fox behind a house, fence, or hedge is hidden.
4. Mischief. The fox scores points for mischief. Ding dong ditch is required: the fox rings a doorbell, the porch light comes on, and the fox has to get clear. Invent at least three more kinds of mischief that suit a fox in a neighborhood.
5. Noise. Mischief draws attention. Kids nearby come to look at where it happened.
6. Five nights. The game is a run of five nights. Each night lasts about two minutes, from dusk to dawn. The fox has a den and must be back in it when dawn comes to keep that night's points. The score carries over from night to night. Three tags in one night end the run.
7. Every night is different. Later nights are harder, and each one adds or changes something: new mischief, more kids, a change in the neighborhood.
8. Story. A short opening that says why the fox is out, a short scene between nights, and an ending that reflects the whole run: how many nights, how much mischief, how many tags, whether the fox made it through all five.
9. Screens. A title screen that shows the controls, the game itself, an ending screen with the score, and a way to play again without reloading the page.
10. Controls. Keyboard only.

Everything else is yours to decide: the look, the characters, the map, the sound, how being spotted feels, the tone of the writing. Keep it playful and suitable for all ages. A first-time player should get through the first night, and finishing all five should take some skill. I care more about a game that is fun and has personality than about a long feature list.

TECHNICAL RULES

- Write it in C with raylib 5.5 and compile it to WebAssembly with Emscripten 3.1.74. Use no other libraries.
- Both are installed. emcc is on the PATH. The raylib headers are in /opt/raylib/include and the web build of the library is /opt/raylib/lib/libraylib.a. It was built with raylib's default web settings (OpenGL ES 2.0, so shaders are GLSL ES 1.00).
- No asset files. No images, models, fonts, or audio files, and nothing downloaded at build time or run time. Make every shape, texture, and sound in code.
- Put the finished game in dist/. Opening dist/index.html from a plain static file server (for example: python3 -m http.server) in current desktop Chrome must load and run the game.
- Add a build.sh in the top folder that rebuilds dist/ from source with one command.
- It should run smoothly on an ordinary laptop.

CHECK YOUR WORK

Headless Chromium and Playwright for Python are installed. Use them to load the game, take screenshots, and read the browser console. Build it, run it, look at it, and fix what you find before you stop.

When you finish, write a short README.md: how to play, what mischief exists, what changes from night to night, and anything you want me to know.
```