Bun is joining Anthropic

Jarred Sumner · December 2, 2025

TLDR: Bun has been acquired by Anthropic. Anthropic is betting on Bun as the infrastructure powering Claude Code, Claude Agent SDK, and future AI coding products & tools.

What doesn't change:

- Bun stays open-source & MIT-licensed
- Bun continues to be extremely actively maintained
- The same team still works on Bun
- Bun is still built in public on GitHub
- Bun's roadmap will continue to focus on high performance JavaScript tooling, Node.js compatibility & replacing Node.js as the default server-side runtime for JavaScript

Claude Code ships as a Bun executable to millions of users. If Bun breaks, Claude Code breaks. Anthropic has direct incentive to keep Bun excellent.

What changes:

- We will help make coding tools like Claude Code & Claude Agent SDK faster & smaller
- We get a closer first look at what's around the corner for AI coding tools, and make Bun better for it
- Bun will ship faster.

How Bun started

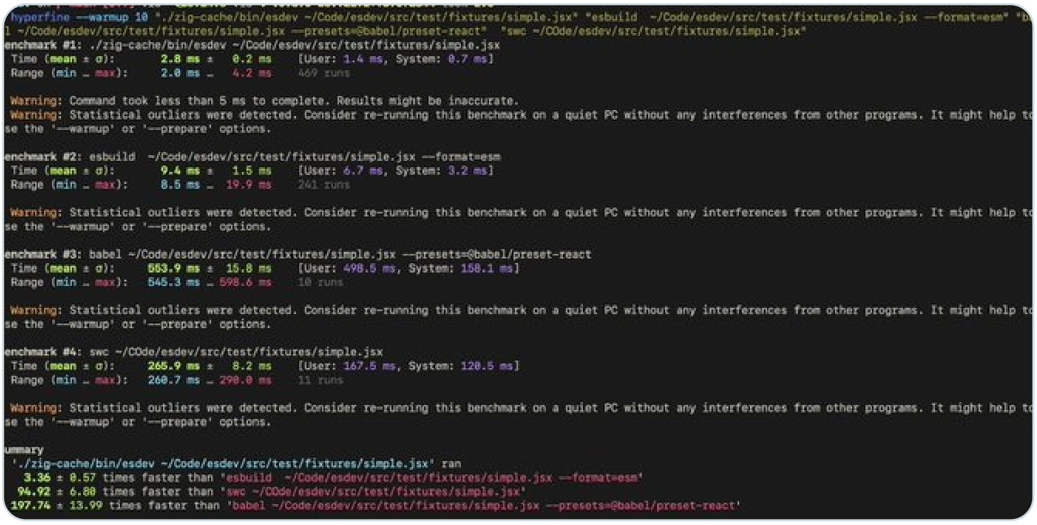
Almost five years ago, I was building a Minecraft-y voxel game in the browser. The codebase got kind of large, and the iteration cycle time took 45 seconds to test if changes worked. Most of that time was spent waiting for the Next.js dev server to hot reload.

This was frustrating, and I got really distracted trying to fix it.

I started porting esbuild's JSX & TypeScript transpiler from Go to Zig. Three weeks later, I had a somewhat working JSX & TypeScript transpiler.

Jarred Sumner
@jarredsumner · **Follow**

Early benchmark from a new JavaScript bundler. It transpiles JSX files:
- 3x faster than esbuild
- 94x faster than swc
- 197x faster than babel



```
 hyperfine --warmup 10 "./zig-cache/bin/esdev ~/Code/esdev/src/test/fixtures/simple.jsx" "esbuild  ~/Code/esdev/src/test/fixtures/simple.jsx --format=esm" "b
l ~/Code/esdev/src/test/fixtures/simple.jsx --presets=@babel/preset-react"  "swc ~/COde/esdev/src/test/fixtures/simple.jsx"
enchmark #1: ./zig-cache/bin/esdev ~/Code/esdev/src/test/fixtures/simple.jsx
 Time (mean ± σ):       2.8 ms ±   0.2 ms    [User: 1.4 ms, System: 0.7 ms]
 Range (min … max):     2.0 ms …   4.2 ms    469 runs

 Warning: Command took less than 5 ms to complete. Results might be inaccurate.
 Warning: Statistical outliers were detected. Consider re-running this benchmark on a quiet PC without any interferences from other programs. It might help to
se the '--warmup' or '--prepare' options.

enchmark #2: esbuild  ~/Code/esdev/src/test/fixtures/simple.jsx --format=esm
 Time (mean ± σ):       9.4 ms ±   1.5 ms    [User: 6.7 ms, System: 3.2 ms]
 Range (min … max):     8.5 ms …  19.9 ms    241 runs

 Warning: Statistical outliers were detected. Consider re-running this benchmark on a quiet PC without any interferences from other programs. It might help to
se the '--warmup' or '--prepare' options.

enchmark #3: babel ~/Code/esdev/src/test/fixtures/simple.jsx --presets=@babel/preset-react
 Time (mean ± σ):     553.9 ms ±  15.8 ms    [User: 498.5 ms, System: 158.1 ms]
 Range (min … max):   545.3 ms … 598.6 ms    10 runs

 Warning: Statistical outliers were detected. Consider re-running this benchmark on a quiet PC without any interferences from other programs. It might help to
se the '--warmup' or '--prepare' options.

enchmark #4: swc ~/COde/esdev/src/test/fixtures/simple.jsx
 Time (mean ± σ):     265.9 ms ±   8.2 ms    [User: 167.5 ms, System: 120.5 ms]
 Range (min … max):   260.7 ms … 290.0 ms    11 runs

 Warning: Statistical outliers were detected. Consider re-running this benchmark on a quiet PC without any interferences from other programs. It might help to
se the '--warmup' or '--prepare' options.

ummary
 './zig-cache/bin/esdev ~/Code/esdev/src/test/fixtures/simple.jsx' ran
   3.36 ± 0.57 times faster than 'esbuild  ~/Code/esdev/src/test/fixtures/simple.jsx --format=esm'
  94.92 ± 6.80 times faster than 'swc ~/COde/esdev/src/test/fixtures/simple.jsx'
 197.74 ± 13.99 times faster than 'babel ~/Code/esdev/src/test/fixtures/simple.jsx --presets=@babel/preset-react'
```

8:22 AM · May 6, 2021

1.3K   Reply   Copy link

Read 36 replies

I spent much of that first year in a very cramped apartment in Oakland, just coding and tweeting about Bun.

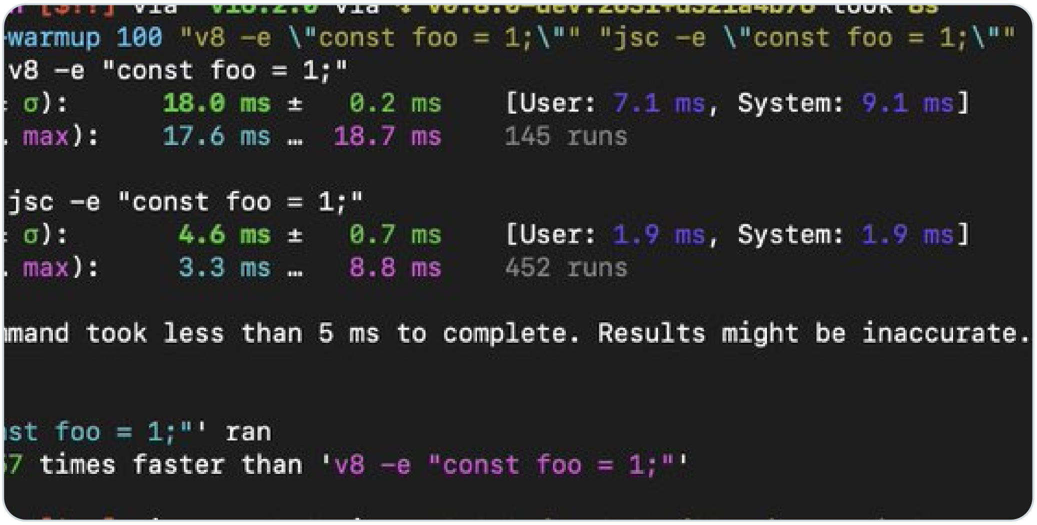
The runtime

To get Next.js server side rendering to work, we needed a JavaScript runtime. And JavaScript runtimes need an engine to interpret & JIT compile the code.

Jarred Sumner
@jarredsumner · **Follow**

The start time difference between JavaScriptCore and V8 is interesting. JavaScriptCore seems to start around 4x faster.

It's possible this is due to the specifics of their respective CLIs though (rather than about JavaScript execution)



```
warmup 100 "v8 -e \"const foo = 1;\"" "jsc -e \"const foo = 1;\""
v8 -e "const foo = 1;"
 σ):      18.0 ms ±   0.2 ms    [User: 7.1 ms, System: 9.1 ms]
 max):    17.6 ms …  18.7 ms    145 runs

jsc -e "const foo = 1;"
 σ):       4.6 ms ±   0.7 ms    [User: 1.9 ms, System: 1.9 ms]
 max):     3.3 ms …   8.8 ms    452 runs

mand took less than 5 ms to complete. Results might be inaccurate.


st foo = 1;"' ran
7 times faster than 'v8 -e "const foo = 1;"'
```

4:38 AM · May 27, 2021

28 Reply Copy link

Read more on X

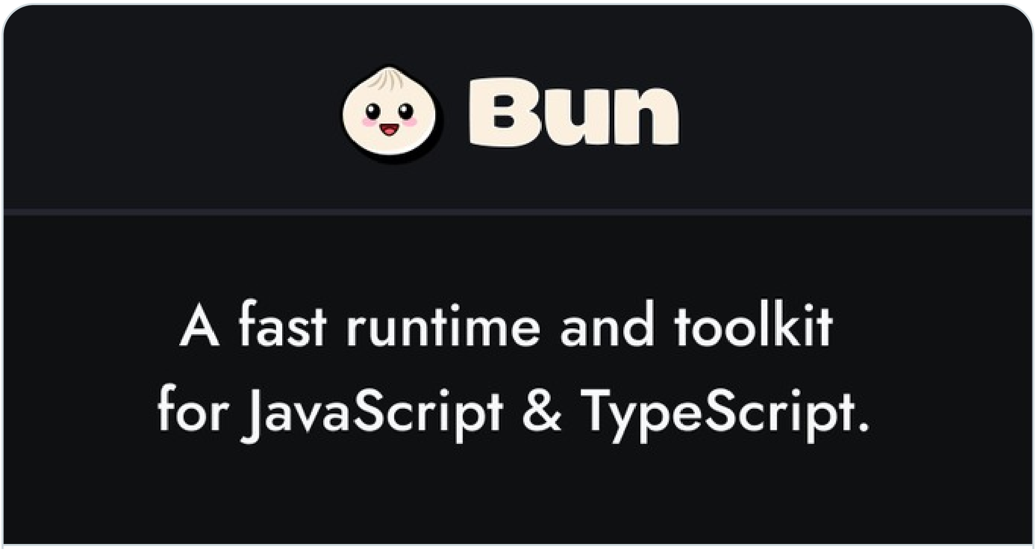
So after about a month of reading WebKit's source code trying to figure out how to embed JavaScriptCore with the same flexibility as what Safari does, I had the very initial version of Bun's JavaScript runtime.

Bun v0.1.0

Bun v0.1.0 was released in July of 2022. A bundler, a transpiler, a runtime (designed to be a drop-in replacement for Node.js), test runner, and a package manager - all in one. We ended up reaching 20k GitHub stars in the first week.

Jarred Sumner
@jarredsumner · Follow

Introducing Bun - an incredibly fast all-in-one JavaScript runtime.



bun.sh
Bun — A fast all-in-one JavaScript runtime
Bundle, install, and run JavaScript & TypeScript — all in Bun. Bun is a new JavaScript runtime with a native bundler, transpiler, task runner, and np...

8:19 AM · Jul 6, 2022

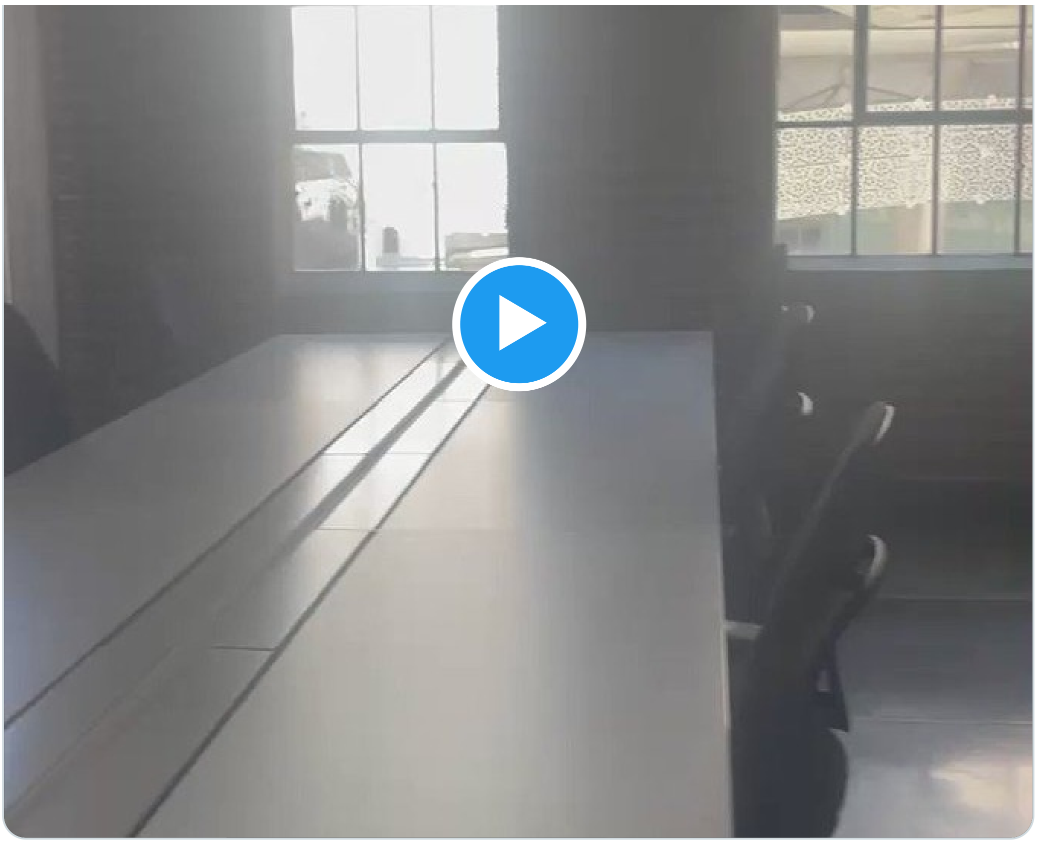
7.2K Reply Copy link

Read 224 replies

Those first two weeks after the release were one of the craziest weeks of my life. My job switched from writing code all day to replying to people all day. We raised a $7 million seed round led by Kleiner Perkins (thanks Bucky & Leigh Marie! And also Shrav Mehta), I took a salary and convinced a handful of engineers to move to San Francisco and help build Bun.

Jarred Sumner
@jarredsumner · Follow

got keys & desks for oven’s office today



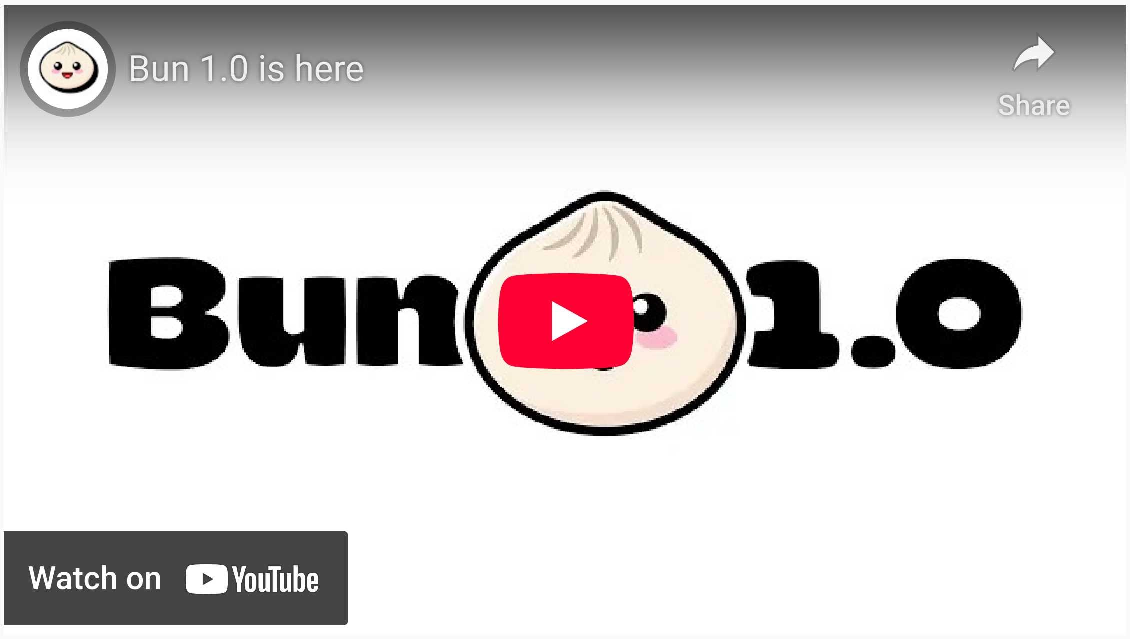
5:34 PM · Oct 8, 2022

290 Reply Copy link

Read 15 replies

Bun v1.0.0

Bun started to feel more stable, so we shipped Bun v1.0 in September of 2023.



Production usage started to pick up and we raised a $19 million Series A led by Khosla Ventures (thanks Nikita & Jon!), grew the team to 14 people and got a slightly larger office.

Bun v1.1

After all this time, we still didn't have Windows support. And every day, people asked us the same question: "when will Bun support Windows?"

So we added Windows support and called it Bun v1.1. Our Windows support was pretty rough at first, but we've made a lot of progress since then.

Bun v1.2

Bun v1.2 made big improvements to Node.js compatibility, added a builtin PostgreSQL client and S3 client. We also started seeing production usage from companies like X and Midjourney. Tailwind's standalone CLI is built with Bun.

Bun v1.3

Bun v1.3 added a builtin frontend dev server, a Redis client, a MySQL client, several improvements to `bun install` and improved Node.js compatibility. The real feature: continued increasing production usage.

AI started to get good

In late 2024, AI coding tools went from "cool demo" to "actually useful." And a ton of them are built with Bun.

Bun's single-file executables turned out to be perfect for distributing CLI tools. You can compile any JavaScript project into a self-contained binary—runs anywhere, even if the user doesn't have Bun or Node installed. Works with native addons. Fast startup. Easy to distribute.

Claude Code, FactoryAI, OpenCode, and others are all built with Bun.

I got obsessed with Claude Code

I started using Claude Code myself. I got kind of obsessed with it.

Over the last several months, the GitHub username with the most merged PRs in Bun's repo is now a Claude Code bot. We have it set up in our internal Discord and we mostly use it to help fix bugs. It opens PRs with tests that fail in the earlier system-installed version of Bun before the fix and pass in the fixed debug build of Bun. It responds to review comments. It does the whole thing.

This feels approximately a few months ahead of where things are going. Certainly not years.

The road ahead

Today, Bun makes $0 in revenue.

One of the most common questions I get is about sustainability. Questions like:

> "How does Bun become a business?"

> "If I bet my work project or company's tech stack on Bun, will it still be around in five or ten years?"

Our default answer was always some version of "we'll eventually build a cloud hosting product.", vertically integrated with Bun’s runtime & bundler.

But the world when I first started working on Bun is different from the world today. AI coding tools are this massive change to how developers do productive work, and the infrastructure layer matters more when agents are writing code.

Forcing ourselves down the prescribed path felt wrong when AI coding tools are getting this good, this fast.

The walk

We've been prioritizing issues from the Claude Code team for several months now. I have so many ideas all the time and it's really fun. Many of these ideas also help other AI coding products.

A few weeks ago, I went on a four hour walk with Boris from the Claude Code team. We talked about Bun. We talked about where AI coding is going. We talked about what it would look like for Bun's team to join Anthropic. Then we did that about 3 more times over the next few weeks. Then I did that with many of their competitors. I think Anthropic is going to win.

Betting on Anthropic sounded like a more interesting path. To be in the center of things. To work alongside the team building the best AI coding product.

This is a little bit crazy

At the time of writing, Bun's monthly downloads grew 25% last month (October, 2025), passing 7.2 million monthly downloads. We had over 4 years of runway to figure out monetization. We didn't have to join Anthropic.

Instead of putting our users & community through "Bun, the VC-backed startups tries to figure out monetization" – thanks to Anthropic, we can skip that chapter entirely and focus on building the best JavaScript tooling.

Why this makes sense

When people ask "will Bun still be around in five or ten years?", answering with "we raised $26 million" isn't a great answer. Investors eventually need a return.

But there's a bigger question behind that: **what does software engineering even look like in two to three years?**

AI coding tools are getting really good, really fast and they're using Bun’s single-file executables to ship CLIs and agents that run everywhere.

If most new code is going to be written, tested, and deployed by AI agents:

- The **runtime** and **tooling** around that code become way more important.
- You get a lot more code overall, written & tested a lot faster.
- Humans are more detached from every individual line, so the environment it runs in has to be **fast and predictable**

Bun started with a focus on making developers faster. AI coding tools do a similar thing. It’s a natural fit.

Bun joins Anthropic

So that's why we're joining Anthropic.

Anthropic is investing in Bun as the infrastructure powering Claude Code, Claude Agent SDK, and future AI coding products. Our job is to make Bun the best place to build, run, and test AI-driven software — while continuing to be a great general-purpose JavaScript runtime, bundler, package manager, and test runner.

Being part of Anthropic gives Bun:

- Long-term stability. a home and resources so people can safely bet their stack on Bun.
- A front-row seat to where AI coding tools are headed, so we can shape Bun around that future instead of guessing from the outside.
- **More firepower.** We’re hiring engineers.

And for existing users, the core promise stays the same:

- Bun remains open-source & MIT-licensed.
- Bun is still built in public.
- The same team still works on Bun.
- We’re still obsessed with making JavaScript and TypeScript faster to install, build, run and test.

Anthropic gets a runtime that’s aligned with where software development is going. We get to work on the most interesting version of that future.

This is going to be really fun.

Frequently asked questions

Q: Is Bun still open-source & MIT-licensed?

A: Yes.

Q: Will Bun still be developed in public on GitHub?
A: Yes. We'll still be extremely active on GitHub issues & pull requests.

Q: Does Bun still care about Node.js compatibility & being a drop-in replacement for Node.js?
A: Yes.

Q: Is the same team still working on Bun full-time?
A: Yes. And now we get access to the resources of the world's premier AI Lab instead of a small VC-backed startup making $0 in revenue

Q: What does this mean for Bun's roadmap?
A: Bun's team will be working more closely with the Claude Code team, and it probably will look similar to the relationship between Google Chrome <> V8, Safari <> JavaScriptCore, Mozilla Firefox <> SpiderMonkey, but with more independence to prioritize the wide variety of ways people & companies use Bun today.