Masters of Scale

How investing in fast, semantic code browsing helps Twitter scale engineering productivity

DECEMBER, 2016

Few companies have had the impact Twitter has had since it launched in 2006. The social networking service has been described as the "pulse of the planet," playing a crucial role in just about every culture shift in the last decade. Behind the scenes, Twitter is also an innovator in engineering culture—a fact that becomes even more impressive when you consider that the challenges Twitter faces are formidable.

Scaling 140 characters to 313 million monthly active users.

Twitter's codebase is huge and highly complex. It takes a sophisticated engineering organization to build and maintain a product that supports sharing messages, images, video, and more across a global community of 313 million monthly active users (and counting). To scale its products and infrastructure, Twitter uses a combination of languages, including Java, Scala, and others. In many cases, engineering productivity actually goes down with scale as problems with communication and coordination limit the benefits of collaboration. To combat all of this, Twitter has an entire Engineering Effectiveness department focused on investing in people, processes, and tooling to boost the productivity of every Twitter developer.

The Problem:
Understanding and reusing existing code.

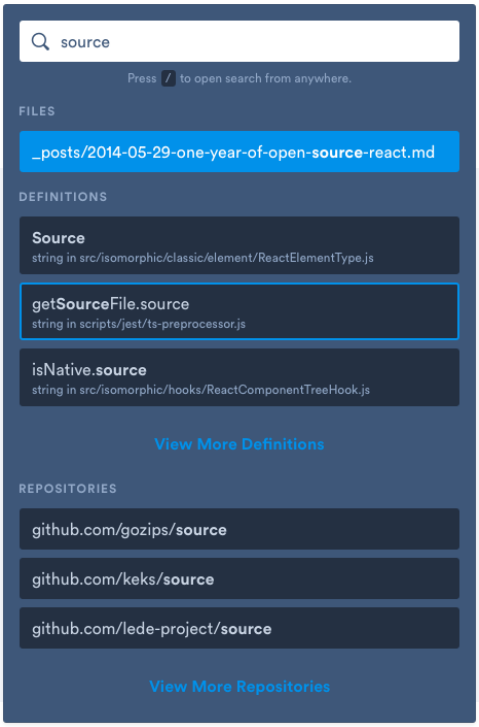
Last year, a small team of engineers in Twitter's Engineering Effectiveness department got together and discussed high-impact ways to improve developer productivity.

The problem? Twitter's codebase was so large and complex that it was hard to understand how each piece of code affected—or was affected by—everything else. Moreover, the existing internal code browser simply couldn't handle the scale of Twitter's codebase. The net result was that navigating the Twitter codebase was slow. And because engineers could not easily answer code-related questions on their own, they often interrupted their teammates with questions, adding to the communication and coordination overhead.

Building a solution in-house was going to take too long and require too big an investment, especially given all the other infrastructure and product priorities at the time. That's when the team, led by veteran engineering director David Keenan, started searching for out-of-the-box solutions.

Sourcegraph met their requirements.

The Solution:
Fast, semantic code browsing.

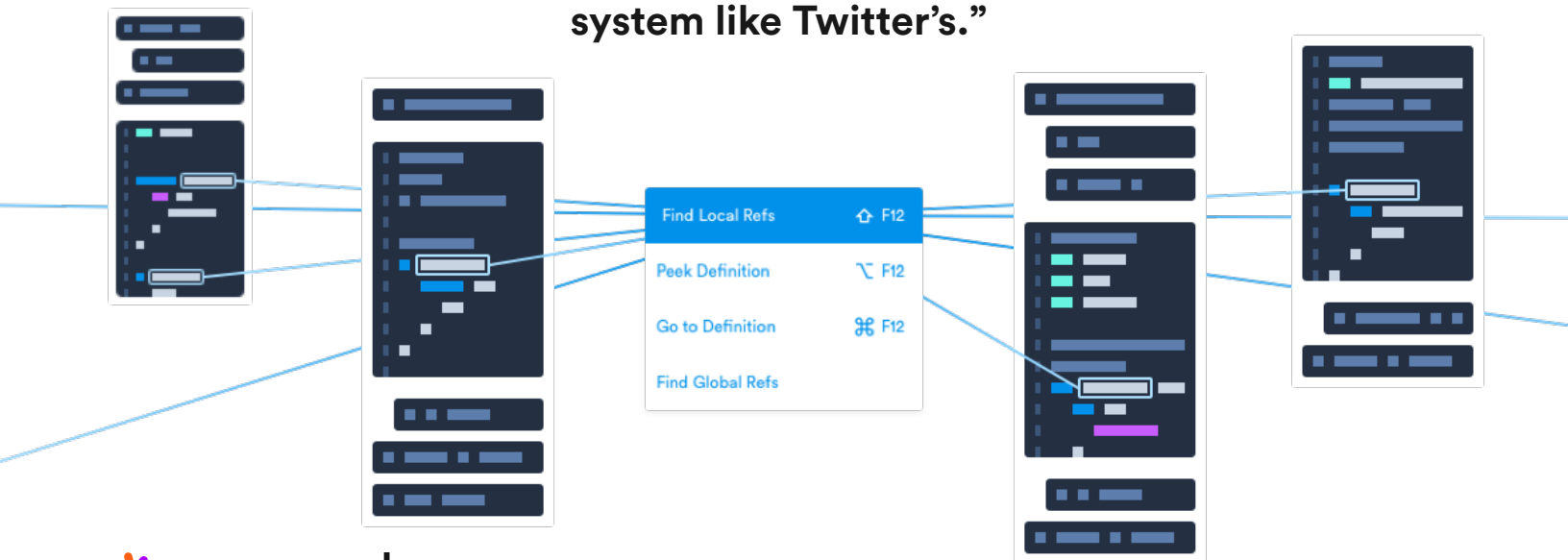


Sourcegraph is a fast, semantic code search and cross-reference engine. It allows users to search for any function, type, or package and see how other developers use it, globally. It's also massively scalable, with 2,000,000,000+ functions in its public code index (and growing).

Under Keenan's leadership, Twitter's team brought Sourcegraph in to boost engineering productivity. They chose Sourcegraph because they believed it could become a go-to resource in their internal suite of developer tools.

During Hack Week, Keenan's team built Scala support on the Sourcegraph API, and the tool was deployed to all of Twitter engineering within a week. "Sourcegraph is easy to integrate into your internal ecosystem because all it needs is a Git clone URL," says Keenan.

"It works even with a completely homegrown repository hosting system like Twitter's."



Sourcegraph indexes the main codebase inside of Twitter and helps developers find the answers they need in seconds, not minutes. It gives them something no IDE can: the ability to easily explore the entire codebase with all its dependencies and discuss code efficiently by linking to specific functions and types.

The Results:
Time saved and limitless potential.

Engineers across many different teams now use Sourcegraph multiple times every week, making it a key part of Twitter's Engineering Effectiveness toolkit. In the words of one senior engineer, "It's very helpful to have functions viewable and clickable in the browser, so you don't have to lose your place in your code editor."

The team saves time in three ways: