DOCKYARD

The Business Value of Elixir

WHY ELIXIR IS THE COST-SAVING, FUTURE-PROOFING, GROWTH-BOOSTING TOOL FOR YOUR MODERN BUSINESS.

Table of Contents

Introduction

Since José Valim **created it in 2012**, the Elixir programming language has been solving problems. At first, it solved the challenges Valim experienced during his time as a Ruby programmer and addressed the need for a language built to handle multicore processors as they became the norm.

Over time, however, those same benefits have expanded to cover a wide swath of use cases. While Valim originally built Elixir to leverage BEAM—**the virtual machine behind Erlang**—to handle the needs of multicore machines, that decision now means that Elixir comes with out-of-the-box capabilities that make it uniquely suited to modern digital products.

Because Erlang was built to seamlessly **handle the demands of telephony systems**, it is purpose-built to handle massive amounts of traffic with as little downtime as possible. Since then, Erlang has become a backbone of the modern internet for its reliability and ability to handle huge amounts of traffic—Cisco estimates that **90% of worldwide internet traffic passes through Erlang-controlled network switches**.

Elixir is built on that foundation. With that comes the same stability, concurrency, and scalability benefits that Erlang was built for, with a programming language that's easy for developers to write and flexible to meet the needs of any number of businesses.

As digital products play an ever-growing role in the day-to-day lives of people around the world, more and more companies will need to deliver products that are reliable, work for huge numbers of people at once, and can grow quickly without missing a beat.

In short, more companies need Elixir.

Reliability

Goods and services, public support, and our work lives are becoming increasingly digital. This means that application reliability is important and will only become more important each year.

When an app fails in production the impact on an organization can be significant:

- Downtime can cost as much as **$9,000 a minute**
- 32% of respondents in one study said they would **abandon an app for a competitor** if they experienced poor performance
- In the same survey, 20% of respondents said they would not trust a company if its app was reported in the media as having poor performance
- 88% of Americans said they had **negative feelings about a brand** if it produced a poorly performing website or app

And the stakes are even higher for some industries: **63% of end users expect their banking and financial services app to work flawlessly**. That means, if your organization is providing a digital financial tool to your users, the margin for error is even narrower than in other sectors, and your brand is on the line every time a user opens your app.

The poor performance of a product or service can not only leave a long-lasting negative impression of your brand, but it also incites users to turn to competitors instead.

This is where Elixir's foundation in Erlang can be a major asset to modern startups and enterprise companies alike. Simply put, Elixir is inherently reliable and results in minimal downtime for your app.

Elixir is built so that, even if one part of an application crashes, the system as a whole can continue running.

This results in more uptime for the application, but also has the important side effect of requiring few lines of code and significantly less development time.

It allows companies to more easily build reliable systems. Something that would require architect-level acumen and a great deal more code and developer time to deliver in other languages is achieved by default with Elixir.

63% OF END USERS EXPECT THEIR BANKING AND FINANCIAL SERVICES APP TO WORK FLAWLESSLY.

Scalability

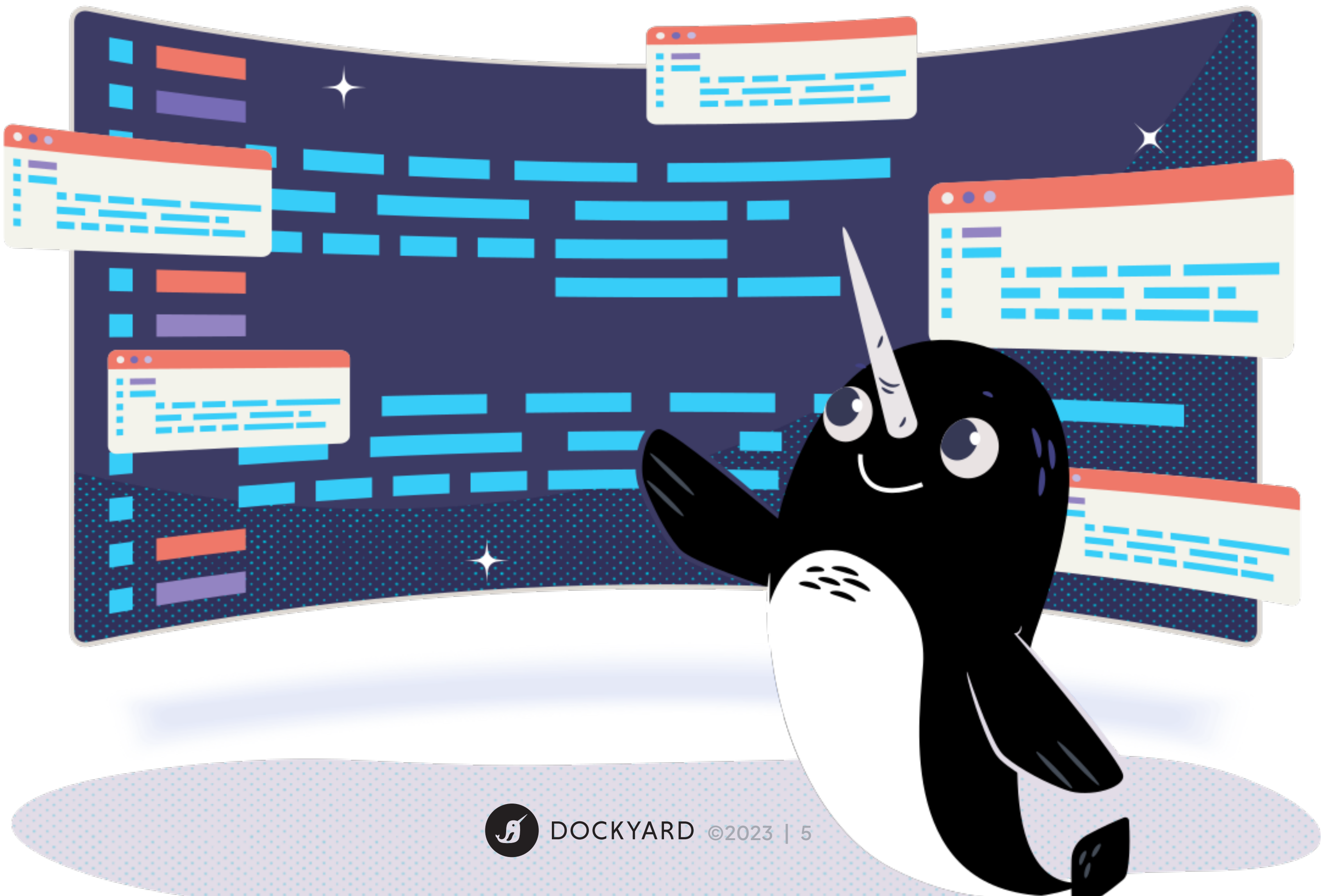
Elixir's out-of-the-box capabilities make it well-suited for early-stage startups preparing for long-term growth.

The ability to build an app that will support the first 1,000 users as easily as the next 1,000,000 has significant implications for startups and market leaders alike.

Take Veeps, a livestreaming concert platform, for example. Originally designed as a platform for artists to sell VIP tour packages, Veeps began offering livestreaming events when COVID-19 brought in-person concerts to a halt. But its Ruby on Rails MVP couldn't handle the traffic, and the platform couldn't keep up with user demands. Add to that a sharp increase in the complexity of Veeps' flagship product, and it was clear that without a dramatic technical shift, Veeps risked losing its newfound user base and the revenue that came with it.

To meet that need, Veeps **chose to move its app entirely to Elixir**. The change allowed it to expand real-time interactions and support by 10 to 100 times. With Elixir, Veeps could easily host the traffic needed to accommodate any artist, no matter how large their fanbase (and, in turn, how large the number of concurrent users watching a livestream).

Elixir's ability to offer scalability of this magnitude not only benefits applications that experience huge surges in traffic but also applications that don't have millions of users but have features that could measure millions of data points. It opens the door to features and functionality that would have previously been impossible or impractical to implement.



Infrastructure Costs

In some cases, throwing additional hardware at the problem fills a temporary need for scalability, but with continued user growth and adoption, that leads to ever-increasing application costs.

Developing applications in more performant languages—like Elixir—however, enables your organization to break this spiraling infrastructure cost dilemma.

With more than 200 million active users, Pinterest, an early Elixir adopter, used the language to rewrite its back-end Java code responsible for managing its rate limit service for both the Pinterest API and Ads API.

The new system's 50% response time is around 500 microseconds with a 90% response time of 800 microseconds. The Elixir system is not just faster and more consistent, but also able to run on half the number of servers the older Java system required. The result is a **$2 million savings** for Pinterest on server costs.

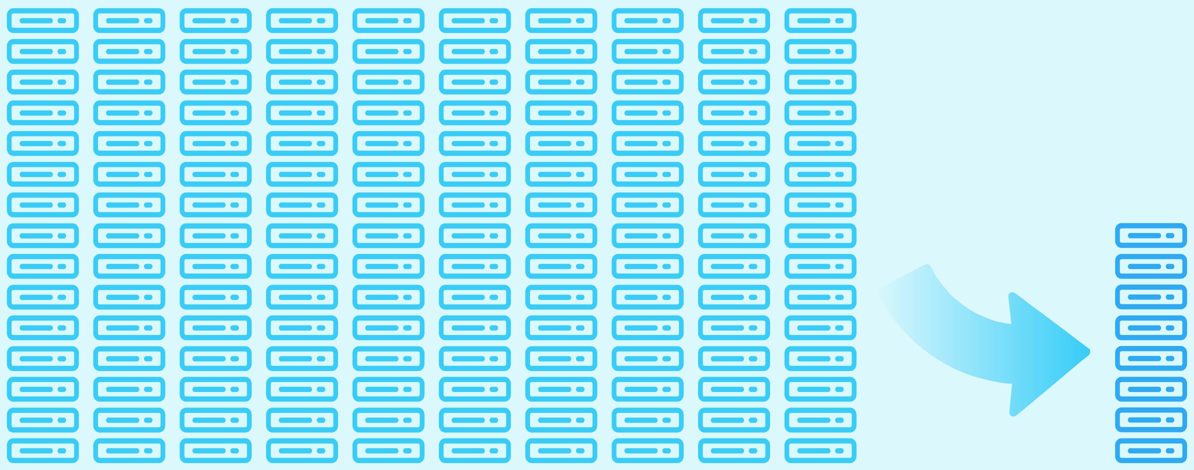
An even more profound example of speed and performance comes from Bleacher Report, the sports culture news organization with over 20 million unique monthly users.

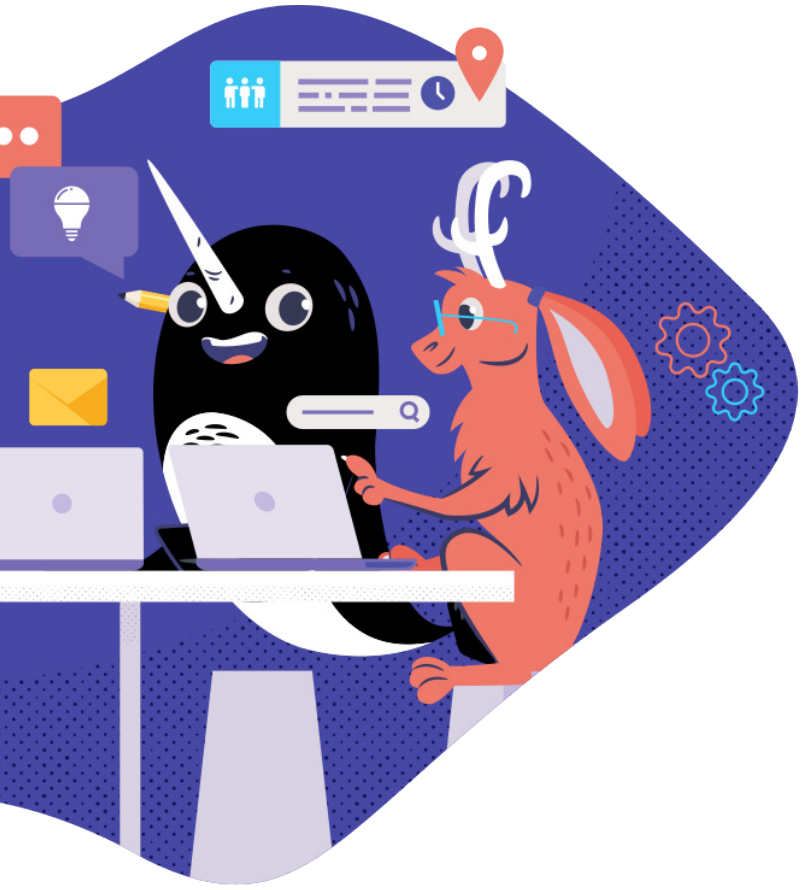
Migrating from Rails to Elixir dropped its number of necessary servers from 150 to just eight, and allowed them to reduce the number of people on their development team.

The migration also led to a significant performance boost and allowed Bleacher Report to send out more than **200 million push notifications** a day **between one and seven minutes faster than its competitors**. In an industry as competitive and reliant on real-time alerts as sports entertainment, that time difference is a significant differentiator for Bleacher Report.

WITH ELIXIR, BLEACHER REPORT DROPPED ITS NUMBER OF NECESSARY SERVERS FROM

150 TO JUST 8





Developer Productivity

One of the less obvious—but remarkably impactful—benefits of Elixir is the new levels of efficiency it introduces for developers.

Studies have shown that developers spend anywhere from 23% to 33% of their time addressing technical debt, and a 2020 report estimated that technical debt cost U.S. companies about $1.3 trillion.

Elixir helps mitigate that cost for your business. With efficient, clear code that avoids the complications that lead to technical debt in other languages, your team will spend less time addressing technical debt and more time working on revenue-generating projects. Oftentimes companies gravitate toward their historic competencies without measuring the economic impact of alternative solutions.

Compare Elixir to a hypothetical Rails app, for example. A Rails app might depend on several distinct pieces. You can only handle one web request at a time and can't spin up new processes on demand, so the solution is to use tools that spawn multiple application servers upfront and put a web server in front to hand off requests.

You can't take care of slow background tasks without blocking your web requests, though, so you use Redis and Resque to handle the background tasks. You also can't keep a large Ruby process running all the time to do scheduled tasks, so you add a dependency cron. And since there's no built-in Ruby tool for managing multiple parts of a running system, you turn to yet more tools to start and monitor them.

Our hypothetical Elixir app needs none of that. Your team gets the functionality they need for a stable app out of the box. That means fewer things to manage—like all the stopgap tools in our hypothetical Ruby app—and fewer things that can break and rack up technical debt.

Due to Elixir's syntax, it is easy to write clean and understandable code without sacrificing developer productivity.

As organizations are challenged to do more with less, Elixir offers a significant opportunity to address a never-ending product backlog. Bottom line: If your development team is spending valuable time fixing technical defects or optimizing for performance, that also means they're spending less time on developing new features or innovating.

“We’re here to tell you that it scales to the real world. You can take a team or a company who doesn’t know anything about Elixir and get them productive in a few short months.”

— DANIEL MACKEY, CARS.COM ENGINEERING MANAGER (2019)

Hiring & Talent

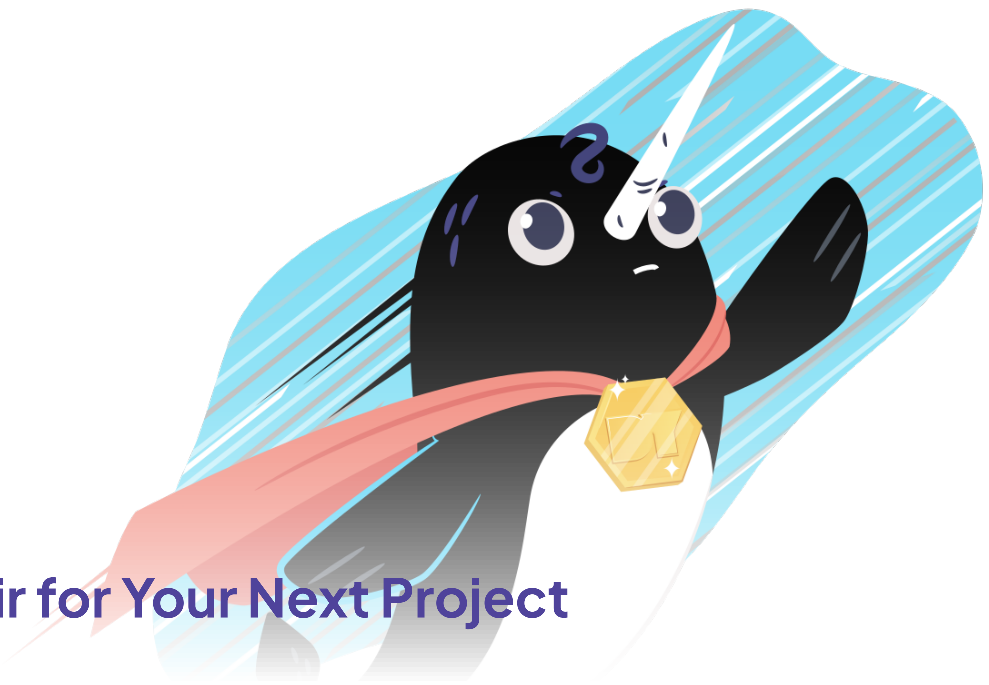
One of the common challenges startup founders and technical leaders cite as a reason not to use Elixir is the comparatively small pool of developer talent and difficulty hiring. However, that’s not the roadblock some may think it is.

On one hand, recent initiatives—like the DockYard Academy—are actively expanding the number of developers in Elixir. The Elixir community is growing and attracting new talent every day.

Aside from those opportunities, developers with experience in another language often find it easy to pick up Elixir as well. That means—even if your development team is composed of engineers with no current Elixir experience—training them on Elixir can be an easy, productive process.

Cars.com is an ideal example of this process. After deciding to migrate to Elixir for an entirely new back-end enterprise architecture, Cars.com used mob coding sessions where an Elixir engineer wrote code live with the Cars.com development team watching to teach them everything they needed to know to maintain and grow the new Cars.com app.

The result was that Cars.com got all the business benefits of an Elixir app, without having to hire a new team of Elixir developers to maintain or grow it. And the same process can work seamlessly for myriad other businesses.



Evaluating Elixir for Your Next Project

Elixir can drive organizational value based on its performance, stability, and scalability paired with clean syntax and code clarity. And the fact that the capabilities of Elixir are quickly expanding to include everything from machine learning to native app development and even an Elixir-based CMS, and it's clear that investing in Elixir is the long-term solution that sets your organization up for success now.

As you analyze your next set of application requirements, Elixir should be front-and-center for any greenfield project, but especially for use cases involving any or multiple combinations of the following:

- High availability requirements with little to no downtime
- High scaling needs with minimal budgets
- A need to quickly develop a production-ready MVP
- Systems expecting high traffic
- Web applications requiring fast & consistent response times
- Distributed systems with demanding concurrency requirements (e.g. blockchain)
- Real-time updates (e.g. stock ticker)
- Bidirectional real-time communication with WebSockets (e.g. chat, games)

Other languages can address one or more of the needs above, but there are always trade-offs. Elixir strikes the right balance to leave you with a more reliable, growth-ready application that will meet your users' needs for years to come.

About DockYard

DockYard is a digital product consultancy. We specialize in production-ready apps that scale as companies grow.

For over a decade, companies like Apple and Netflix have trusted us to overcome complex product challenges. We help to upskill teams through a range of consulting services with capabilities in product design, full-stack engineering, project management, QA, strategy, training, and user experience (UX).

We're also dedicated to advancing open-source web development technologies, such as libraries and tooling built around the Elixir programming language. From idea to impact, DockYard empowers forward-thinking teams to build for the future. Ready to put our expertise to work for you? Contact us at hello@dockyard.com.

DockYard's Commitment to R&D

$7.3M in R&D

INVESTED OVER THE PAST SEVEN YEARS TO ADVANCE THE WEB FOR ALL

+2,000 hours

DEVOTED TO PROGRESSING OPEN-SOURCE PROJECTS ANNUALLY

Countless Innovators

USING "DOCKYARD DAYS" TO GROW AND CONTRIBUTE TO DIGITAL PRODUCT COMMUNITIES

DOCKYARD