EmberJS Homepage Survey results

At the end of July, we published a survey asking Ember users to talk about why they continue to use Ember in a world full of alternatives.

Below we've summarized the ***Values*** portion of the survey. We think you'll find the responses encouraging – there's a lot of great stuff said about Ember!

Whenever a developer says "I want to…", **there's usually a powerful statement that follows**.

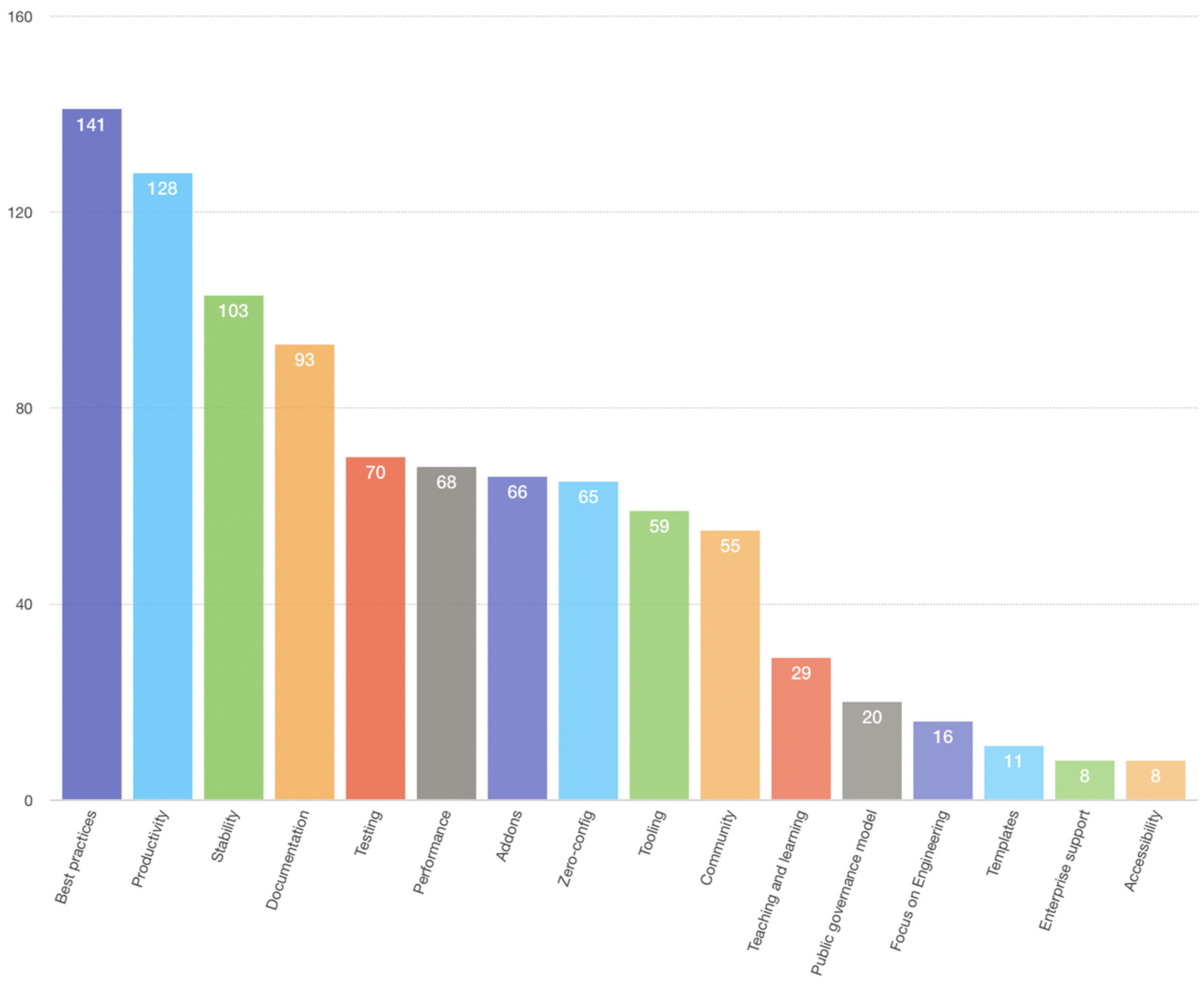
We think these results should influence the content and emphasis of the new homepage – perhaps we can even use some of the quotes there!

Values of Ember developers

The question was:

> *Which of these values are most important to you as an Ember developer? (Please choose up to 4.)*
>
> *Can you elaborate on what you had in mind when making your choices?*

Here are the totals:



Clear best practices and **Productivity** came out in front.

For each value, we've selected some highlights from the results and listed them below. Use the survey ID at the end of a quote if you'd like to look up the full result in Wufoo.

Dropbox Paper has a nice autogenerated nav on the left-hand side if you'd like to jump around sections.

Clear best practices

Best-practices and zero-config helped us "get the job done". We didn't need to explore thousands of different approaches and try them out for weeks. (#126)

Clear best practices means less bike-shedding about "how" we're going to do something, and more focus on "what" we're doing – and writing the business logic to get that done. (#134)

I want to get creating. I don't want to adjust a thousand cogs just to get ESLint or Babel running. I just want to create, and I want to do it now. (#145)

To me, clear best practices means having a clear way to do something. If you need auth, then you can lean on Ember Simple Auth. If you need X, there's a clear way to do it. Many choices are already made for you, and you can rely on the people making those choices. (#181)

We're a small team that has big ideas, and having a framework that has clear best practices is invaluable. We can focus on our work, instead of figuring out how to setup and connect various libraries. (#185)

To me, the biggest advantage Ember has over other frameworks is its focus on clear, best practices. Other frameworks require that you make many decisions before starting, but Ember leads you down an established path. (#186)

To me, clear best practices mean having shared solutions to the problems I'm most likely to encounter while building a web product. (#197)

The best ideas either start or end up in Ember. (#202)

Ember's "clear best practices" has saved our team a lot of time learning the framework, and lets us easily share and understand each other's code. (#210)

Clear best practices means there's not a "flavor of the month" routing engine or data layer. And when there are new patterns, they're strongly vetted and accompanied by a migration path, often with a codemod. (#206)

Clear best practices helps enforce similar coding styles and strategies across teams. Together with "zero-config," it significantly reduces the time needed to get familiar with existing codebases. This is especially important to me as a contractor, since I switch teams a lot. (#178)

I don't have to waste time figuring out the best way to do things. (#158)

Clear best practices helps me not reinvent the wheel. (#164)

Clear best practices solves the "unknown unknown" problems – the ones I can't reasonably expect to know I'll encounter. Best practices codifies our collected wisdom, and helps me avoid problems I'm not even aware of. (#172)

To me, clear best practices means I don't have to search the Internet to answer my engineering questions. The answers are there right in front of me, in the form of framework-ingrained conventions. (#117)

Clear best practices make it easy to find where things are, even if you didn't write the code. The patterns are easily discernible. (#111)

I appreciate that in Ember apps, there is usually one *and only one* right way to do things. The community standardizes on common idioms for solving complex problems. (#103)

Clear best practices reduce the thought needed to evaluate different ways of doing things. Less decision making about small problems allows me to concentrate on the bigger picture. (#86)

Clear best practices means you don't take wrong turns early on that will require big refactorings later, when you realize what the best practice actually was. (#80)

Best practices are important because we want every new engineer to get up and running quickly. Our team is expanding, and I want everyone to be able to quickly understand exactly what Ember can do. (#76)

Best practices means my team doesn't waste time bike shedding. (#1)

If you know Ember, you can jump into any Ember project and hit the ground running. (#8)

Compared to other JS frameworks out there, Ember's main advantage is the avoidance of bike-shedding at every decision point while building an application – file directory structure, testing approach, build tools, etc. It's all decided for you, and the rest of the community is working within the same configuration that you are. (#10)

Productivity

Productivity for me is about skipping the boilerplate, and getting going on my project quickly. (#160)

“Productivity” captures the core values that are important to me. With Ember I can build "product", not "software". (#106)

Ember allows me to focus on domain problems. That's why we create software. (#122)

I deploy to production 10 times per day, and I feel confident everything works. (#119)

I want the majority of the code I write to be business logic, not framework logic. (#50)

Productivity is the most important part of Ember – from zero configuration, to the Addon ecosystem, to the CLI. You can get something started very quickly, but that doesn't prevent you from expanding that base into something much larger. (#78)

I want to just write code, and be productive when switching between projects. Productivity goes hand-in-hand with conventions, for us. (#77)

Productivity means I have to build out less boilerplate and make fewer choices upfront. I can focus on the really interesting problems in my domain. (#62)

Any developer can jump in on an Ember project, and know how everything is structured right away. (#20)

Stability

Stability means that I started the project that powers my startup on Ember 1.13 and I'm now on Ember 3.2 (#130)

The major-release process, which simply removes existing deprecations, makes it easy to stay up-to-date. We've been using the framework for 5 years and are currently enjoying 3.2. (#133)

Semver, deprecations, LTS versions, and Ember CLI Update === painless upgrades. (#149)

I got burned by the Angular -> Angular N changes. Ember's stability and commitment to semver gives me peace of mind in the long term. (#181)

I won't have to rewrite everything every 6 months. (#203)

I don't have to worry about painful version upgrades. (#158)

Stability without stagnation is a great selling point to me. I work on some apps that only get attention a few weeks a year – stability (together with codemods) lets me keep them up to date without wasting time on maintenance tasks. (#178)

I would not choose a framework that made regular breaking changes, period. The fact that Ember has an LTS model, that upgrading Ember apps these days is usually a simple painless PR, and that the ecosystem generally has support for the previous major version are all huge selling points to me. (#138)

To me, stability means backwards compatibility and being able to upgrade between major versions. No one wants to be left hanging on an old version of a technology. (#139)

Stability is pretty straightforward. I've worked on Ember apps that were started in the days of Backbone and are still performing beautifully today - no big bang rewrites required. (#172)

Stability without stagnation gives confidence to businesses and developers that they will be able to keep their apps innovative and competitive for the long term in a sustainable manner. Choosing Ember will never lock you out of exciting new paradigms (GraphQL, Redux); in fact it may well allow you to adopt them more easily (ember-apollo-client, ember-redux). (#123)

Almost all of the above choices are important to us, but the fact that we can keep working on making our user experience better without the churn of a rewrite, while continuing to get performance and modern JS features in each update, is the number one benefit of Ember for our team. (#118)

We chose Ember for the long run. Our application is 13 years old, and when we started our re-write in 2015 we wanted a framework that would be around for 5 years – maybe even 10. (#118)

Stability means I can spend more time making features, and less time babysitting framework upgrades. (#111)

Stability is something really valuable in this ever-moving JavaScript world. The versioning strategy of major versions only cleaning deprecated code is really inspiring. (#89)

I want to be able to use new features without constantly having to rewrite old code. I want to have clear migration paths when things change. (#77)

Stability means I can build something without redoing everything in a year or two. (#24)

I want the application my company builds to last without a yearly rewrite. (#33)

Testing

I won't lose any time setting up a test harness. (#125)

We have a test-driven mentality, so Ember's clear focus on making testing a first-class citizen in the framework is comforting. (#133)

Testing is easy and encouraged, and it’s not a burden (#177)

Having one way to test and a comprehensive testing library built into the framework is a huge benefit. (#181)

Testing should never feel like a chore. Having a trusted test suite that operates at every useful level unlocks speed and confidence in teams of any size. Great testing tools also aid the design of great user experiences. (#123)

As far as I'm aware, no other technology has as good of a testing suite as Ember's. You get unit/rendering/integration/acceptance tests out of the box when you set up a new app – without any configuration. I can't imagine not having something like Ember's acceptance tests. (#139)

Testing is overlooked, and in general kinda sucks on other frameworks. Ember is super winning here. (#48)

Addons

Addons deserve to stand out as one of most important assets of the ecosystem. (#198)

The addon ecosystem is another reason I value Ember. As a developer, being able to use community solutions is amazing. At a talk I gave recently, I said "There's probably an addon for that" - and it's true! There are so many great addons. (#116)

Addons let me leverage plenty of prior art, and not re-invent the wheel poorly. The community can experiment and focus its efforts to agreed-upon approaches. (#62)

Addons built on shared conventions have been a big win in my view. The ability for the community to easily experiment outside of core is great. (#37)

Ember's addon community is amazing. It is vastly superior to other frameworks, in my opinion. You don't get 1000 addons all trying to do the same thing, while only one or two do it well. (#30)

Zero-config

I can onboard a new developer in no time at all. (#125)

I do not need bring my own router, data layer, build pipeline, etc. (#125)

I get a working app out of the box. (#149)

I don't want to spend time configuring my app. Boilerplate with sensible defaults that allows me to get working immediately is brilliant in my book. (#181)

To me, zero-config means I can set up an app with a test harness without setting up any tooling or configuration. (#139)

Zero-config is great, because I can hop right into my terminal, generate a new Ember app, and be up and running very quickly. The CLI is very nice and helps scaffold the app structure. Not having to worry about which test framework I want to pull in, which router I want to use, etc. is also a great benefit to get me going and being productive. (#116)

Zero-configuration tooling and focus on developer experience is one of Ember's core strengths. (#103)

Zero-config allows me to concentrate one some of the more real problems at hand, like the design and flow of the application. (#86)

Minimal config and stability make me come back to Ember every time I need to build something. I don't have to think about re-learning over and over how the project will be setup (#64)

Tooling

There is no other tool that beats Ember CLI. Being able to start a new app and think only about the business features and not how I'm going to minify my code is great. (#159)

I want my tooling to just work, without me needing to spend hours debugging build processes or optimizing something. (#77)

Ember's tooling is great. Using Ember CLI to generate important stubs, install addons and have a ready-to-use build pipeline is just awesome. Tools like Ember CLI Deploy help get my code shipped. The inspector is really awesome. (#102)

We chose Ember because of the tooling and maturity, compared with everything else (#15)

Community

The Community is totally crushing it. Everyone wants to help each other. Love it. (#119)

Community is what keeps Ember growing. More specifically, a diverse community, from diverse backgrounds, with diverse experiences, working on diverse projects, all pushing Ember to innovate and grow in an inclusive manner that is useful for more people. (#123)

Community is the reason I keep developing with Ember. It's not just shared conventions and solutions – it's an extended team that sharpens me as I seek to help others, and also find help when I need it. (#114)

The community is wonderful. My team has attended the past 3 EmberConfs. I've always found that the core teams are very accessible and friendly. I think this really helps set the tone for the larger community. (#37)

Public governance model

Ember’s public governance model is a strength. Lots of big companies use Ember without dominating its direction. (#202)

Public governance is about getting a feeling for what's going on — it gives a feeling of confidence. (#178)