State of XState 2024

Preface

Thank you for downloading my report on the State of XState for 2023. This report details the most important centers of interest shared across the XState community, based on the responses I got to my survey and the interviews I conducted with the people willing to talk about their experience with XState.

Thirty-nine people responded to my survey, which took place between 13 and 30 May 2023. I contacted XState users before creating the survey to gather advice on which topics my final report should focus on. From the invaluable feedback received, I chose twenty questions for the survey. They focus on the subjects that XState developers want to learn more from others in the industry.

I interviewed seven persons who shared their feedback about XState with me. This feedback allowed me to provide real examples for all the topics covered in the report.

I want to thank everybody that took the time to answer my emails, respond to the survey, and get interviewed. It matters a lot to me, so thank you! Their help made this report accurate on the current State of XState.

Summary

In this report, I'll cover five topics that matter to XState developers.

Onboarding new developers

XState reports being hard to learn. Teams employ different methods to teach XState to developers new to state machines. Reading XState documentation, watching video courses, and giving modeling exercises are a few ways to teach XState.

Modeling state machines

Modeling is known to be the most complicated part of creating state machines. Finding how to best solve a problem by modeling a state machine is a skill that takes practice. Sometimes it may also be better to stick to a simple solution using a more scoped tool instead of a state machine.

Types

Types are crucial for building scalable software. XState supports typing machines. Throughout its history, XState has released different ways to type state machines. People are not expecting the same level of type-safety for their state machines, and they have their preferences about the method to achieve it.

Splitting machines

It is not rare for state machines to grow fast in size as adding more and more features. Splitting machines into smaller ones is a way to decouple independent blocks, as we do with separate functions. XStaters adopted several patterns to split their state machines and to be able to maintain complex systems of actors.

Testing and state machines

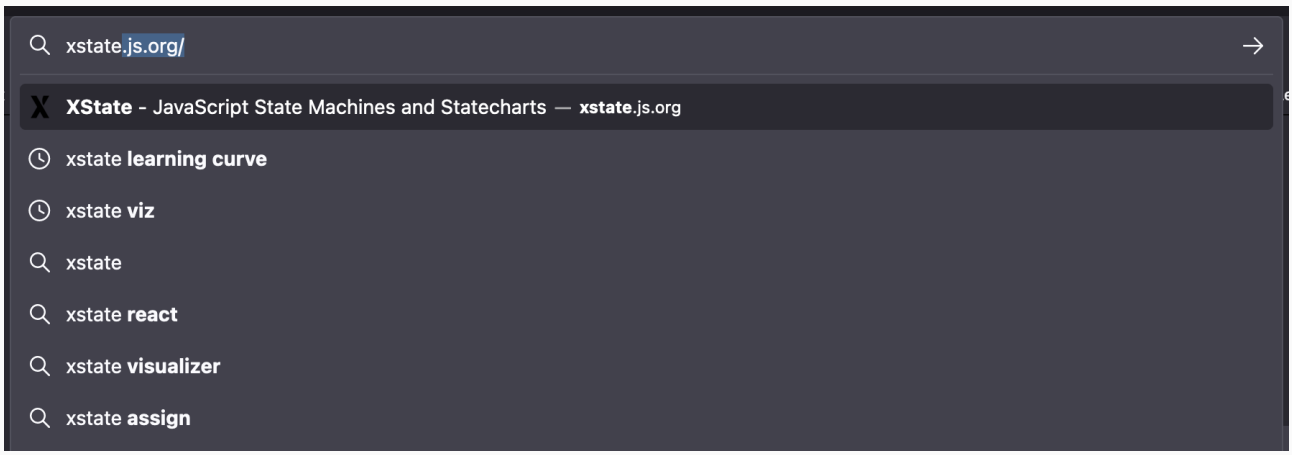
Testing usually leads to divergent opinions and is the same for testing XState machines. Some XStaters choose to unit-test their state machines. Others run integration or e2e tests that don’t care about implementation details. Model-Based Testing is also a pattern used to improve the reliability of software.

Onboarding new developers

When discovering XState, most people also learn the foundational concepts of XState: state machines, finite states, transitions, services, statecharts, hierarchical states, etc. The XState's API is complete and brings tools to solve advanced issues. It takes time to learn the API of the library, but it takes even more time to learn how to use it efficiently and model things.

Many teams worldwide are using XState to develop software, and they are all facing the same problem: how to teach XState to other developers in the team and newcomers? On this aspect, XState does not resemble most of the libraries, like React, TanStack Query, or date-fns, that solve a specific problem and are well-bounded: XState, on its own, does *nothing*; it's a tool that allows expressing any logic declaratively and explicitly, and thus it can be used to do *anything*. The scope of XState is less identifiable, and it can be challenging for people discovering XState to see its benefits.

83% of respondents to the survey declare to ask newcomers to read XState's documentation to learn XState. It has been the official way to learn XState for a long time, and this is how I learned XState. I visited https://xstate.js.org/docs/ so much that my browser now forces me to go there when I type *xstate*. Stately's team is working on updating the documentation, which will now be live at https://stately.ai/docs. Be sure to follow the work they'll be doing there, as it will be amazing! However, my browser will have to learn the new URL of XState's documentation.



My browser forces me to go to XState's documentation when I type `xstate`

One new way to learn XState is through David Khourshid's courses on Frontend Masters. At the time of writing, the most recent courses are about State Machines in JavaScript with XState, and State Modeling in React with XState. 44% of respondents declare they ask newcomers to learn XState with Frontend Masters' courses and others, like Kyle Shevlin's introduction to State Machines with XState. I watched these courses and recommend them too; they are great ways to have an overview of XState's API and modeling techniques, the two difficulties with learning XState.

Another exciting way to learn XState is to understand it based on real scenarios from companies' problems. **Christian Grøngaard** pair-programs with newcomers to teach them how to use XState to solve specific issues, and so do 67% of the respondents. **Richard Seaman** also bets on modeling exercises based on the company's business problems: newcomers have to model with XState a problem they already faced. 25% of the respondents use modeling exercises to teach XState.

Interestingly, no respondent said their team requires XState skills to join it, even if 21% of respondents said they use XState for most of their logic, and 15% use XState everywhere. I'm not sure we would get the same numbers if asking teams developing with React whether React is a required skill. This is mainly because XState developers are rarer than React developers, and training developers in XState is part of the hiring process. But I think it's also due to XState being more versatile and used in many ways to solve various problems. React is mainly used to build web apps and native apps with React Native, though it can also be used to create CLI; XState is mainly used on the frontend — 87% of respondents use XState on the frontend –but it's also used on the backend — 23% of respondents use XState for short-lives machines on the backend, and also 23% of respondents use XState for workflows — and on hardware.

My advice to onboard developers to XState would be to mix reading the documentation with watching courses and applying XState to real problems with your teammates. Most of the respondents combine these ways of learning. If you are learning XState on your side, Stately's community is here to help you on Discord. Learning by example suits many people, and XState Catalogue remains a good reference. It takes time to get good at XState, so take it slow!