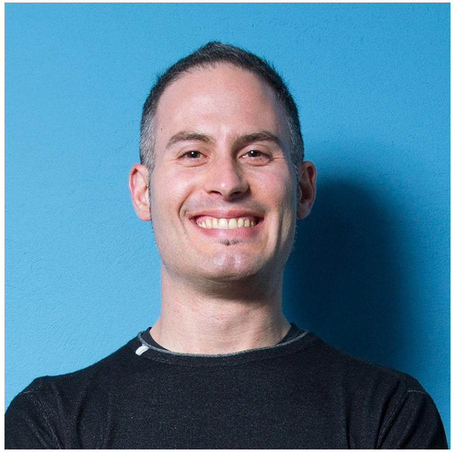
React State Management

A common approach for multiple scenarios

June 2021 / Stefano Magni / @NoriSte

I’m
Stefano Magni



I’m a passionate front-end engineer, a speaker, and an instructor.

I’m the team leader of two frontenders, working at WorkWave RouteManager, a Field Service software.

Table of contents

Current problems with State Managers

- Different apps use different State Managers

- There are multiple State Managers used in the same React app

- We don't agree on a common solution

Why I think they aren't problems

- React State Managers' market evolves quickly
  - We must embrace dynamicity
  - We must be able to move quickly to stay up to date

- Developers like greenfield projects
  - It has an impact on developers' retention
  - It has an impact on the kind of developers we attract and hire

Why I think they aren’t problems

- Every app have different needs
    - Size of the codebase
    - Performance requirements
    - Abstraction level
    - Complexity

- A single tool can’t fit all

We must be open-minded

- We must weigh advantages and disadvantages, not our own preferences
    - Props drilling is not evil in the case of shallow trees
    - React Context is not bad if used where performances aren’t critical
    - Redux offers a great DX if used through Redux Toolkit

My proposal: Decoupling State Managers from the UI

- Doing so, every project looks similar in how it approaches State Managers

- Migrate from one State Manager to another one is easy

- Playing with new State Managers is easy

How?

- By avoiding spreading the implementation details of the State Managers across the app

- By exposing custom hooks and functions that hide the implementation details of the different State Managers

- Working demo: take a look at books and songs in *src/states/*

Sharing takeaways

- For every State Manager, we must document
  - Why we try it
  - What APIs we use
  - Pros and Cons for every use case

- We can compare new State Managers with our tracked use cases

- Sharing our experience with the public attract new developers

There is more to tell

- How to keep different State Managers in sync?
- How to face cases where we can’t completely hide the implementation details of a State Manager?
- Do local states need such boilerplate?

Thank you :)