**What is Redux?**

**Redux is a pattern and library for managing and updating application state, using events called "actions".**

### Why Should I Use Redux?

**The patterns and tools provided by Redux make it easier to understand when, where, why, and how the state in your application is being updated, and how your application logic will behave when those changes occur**.

### When Should I Use Redux?

Redux helps you deal with shared state management.

Redux is more useful when:

* You have large amounts of application state that are needed in many places in the app
* The app state is updated frequently over time
* The logic to update that state may be complex
* The app has a medium or large-sized codebase, and might be worked on by many people

### The Redux Store[​](https://redux.js.org/tutorials/fundamentals/part-1-overview#the-redux-store)

The center of every Redux application is the **store**. A "store" is a container that holds your application's global **state**.

A store is a JavaScript object with a few special functions and abilities that make it different than a plain global object:

* You must never directly modify or change the state that is kept inside the Redux store
* Instead, the only way to cause an update to the state is to create a plain **action** object that describes "something that happened in the application", and then **dispatch** the action to the store to tell it what happened.
* When an action is dispatched, the store runs the root **reducer** function, and lets it calculate the new state based on the old state and the action
* Finally, the store notifies **subscribers** that the state has been updated so the UI can be updated with the new data.

### Data Flow

