A quick guide about **Redux** & Redux Toolkit(RTK) In React / TypeScript

fabiobiondi.dev

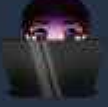
WHAT IS REDUX?

1. **Predictable State Management** by maintaining a single source of truth, it becomes easier to understand and debug how data changes over time. Redux's immutability and **unidirectional data flow** ensure **consistent** and **reliable state management**.

2. **Centralized Store:** this eliminates the need to pass data through multiple components, simplifying data flow and making it easier to access and update the state from anywhere in your app.

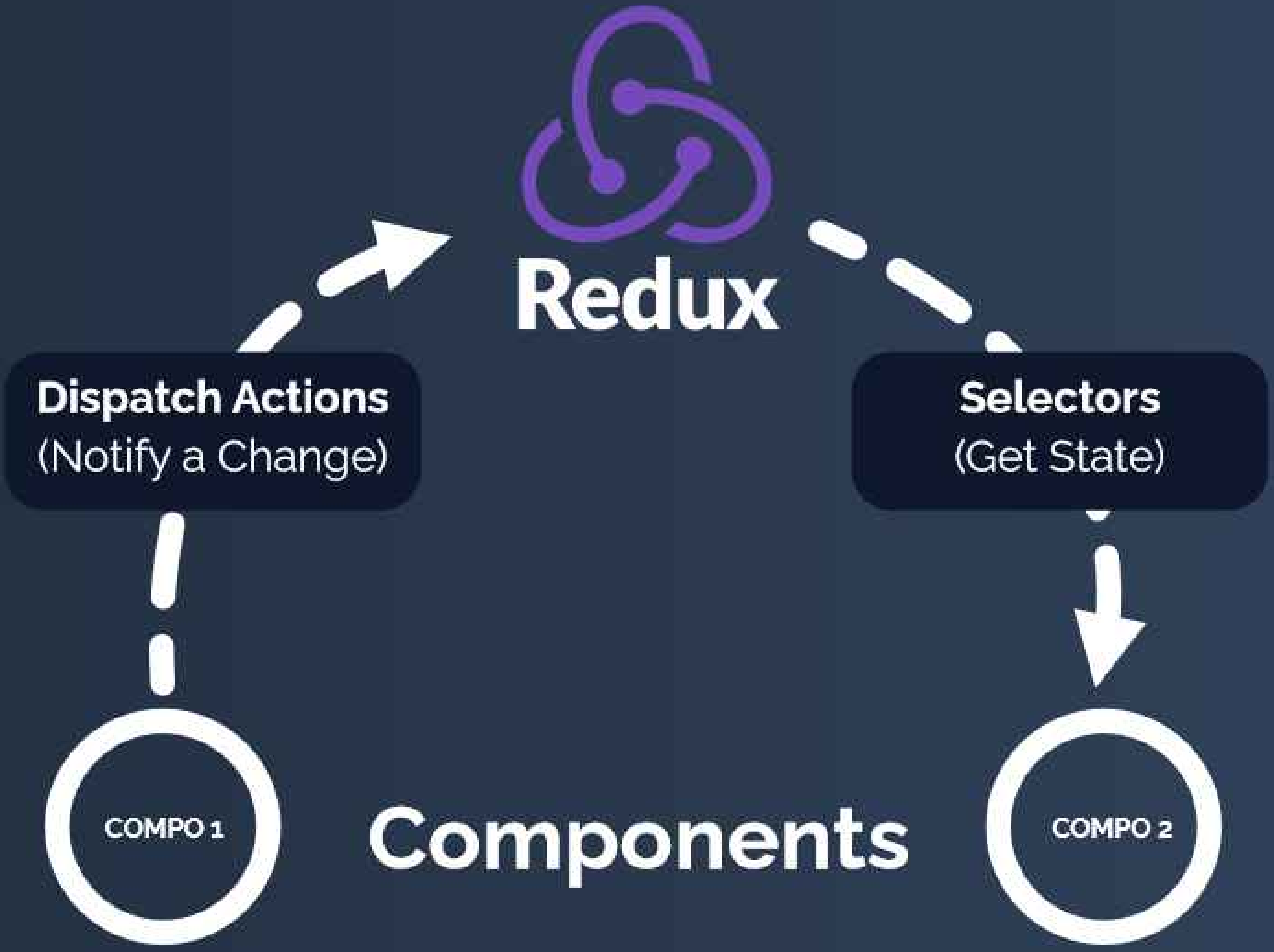
3. **Efficient Updates with Reducers:** Redux utilizes reducers to handle state updates. These **pure functions** provide a clear and structured way to modify the state, making it easier to maintain and reason about changes in your app

4. **Time Travel Debugging:** Redux's **powerful DevTools** extension allows for time travel debugging. You can **replay actions** and **inspect** the **state** at any point in time, making it a breeze to track down bugs and understand how your application's state evolves



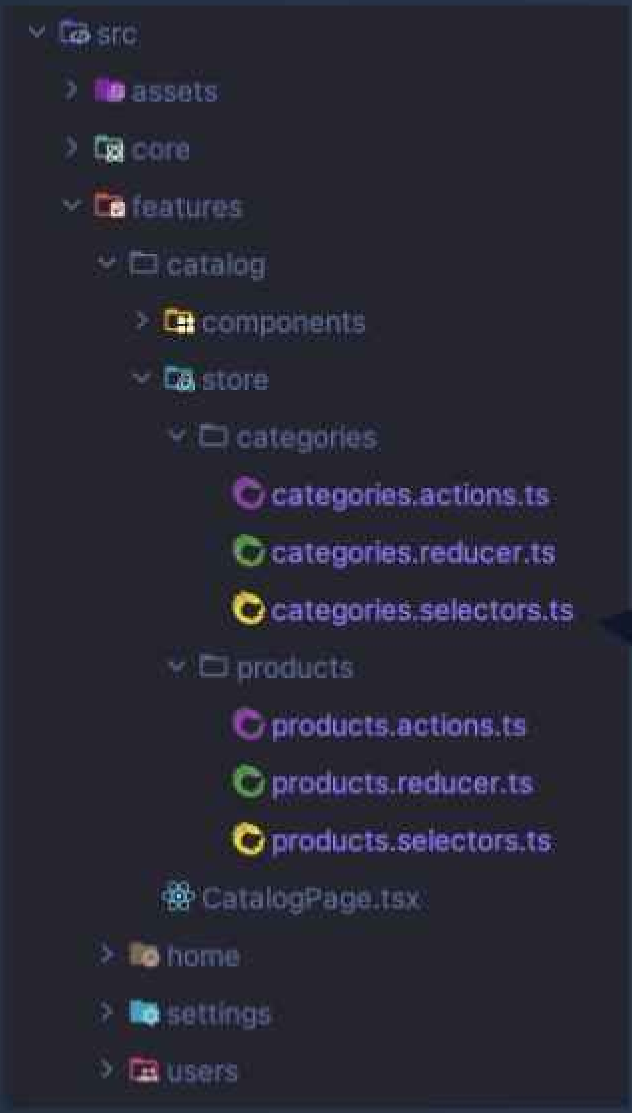
REDUX PRINCIPLES

- **Single Source of Truth**
  (one global state aka STORE)
- **State is read-only**
  (only actions can update the state)
- **Changes are made with pure functions**
  (using reducers)



WHY REDUX?

- State is **centralized** and **global**
- It's a well known **pattern**
- **Opinionated**: it helps developers to have some discipline
- **Don't reinvent the wheel**: for most commod scenarios can easily understand the architecture. Why?
- **Redux Dev Tools** (AMAZING!! See next slides)
- **Avoid unnecessary renders**

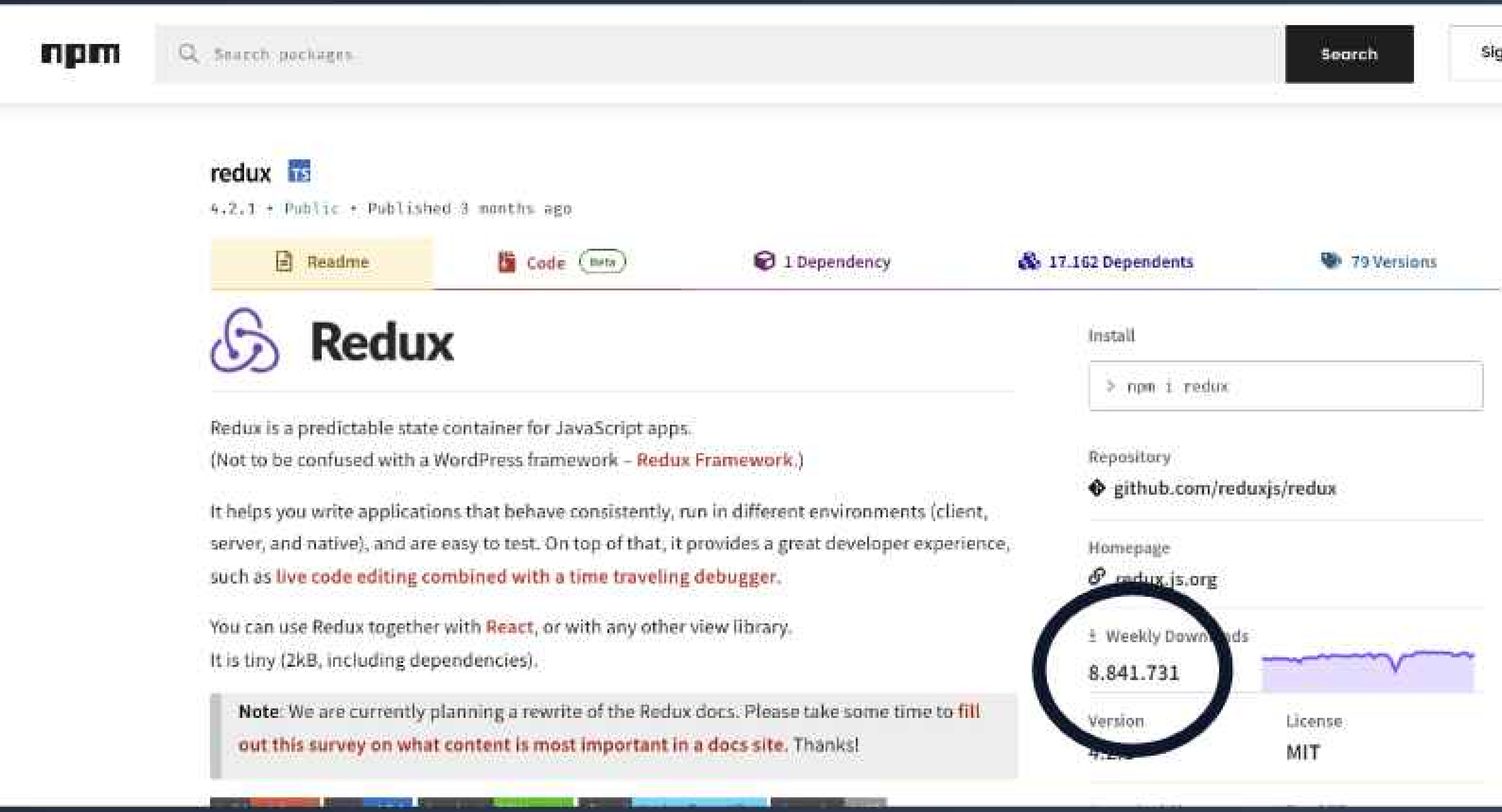


```
src
  assets
  core
  features
    catalog
      components
      store
        categories
          categories.actions.ts
          categories.reducer.ts
          categories.selectors.ts
        products
          products.actions.ts
          products.reducer.ts
          products.selectors.ts
      CatalogPage.tsx
    home
    settings
    users
```

Who uses Redux?

Facebook, Netflix, AirBnb, Uber, Pinterest, Microsoft, GotoMeeting, Atlassian, StackBlitz are some of the companies that are using Redux.

npm 8.8M dowload / week



redux TS
4.2.1 • Public • Published 3 months ago

Readme | Code Beta | 1 Dependency | 17.162 Dependents | 79 Versions

Redux

Redux is a predictable state container for JavaScript apps.
(Not to be confused with a WordPress framework – Redux Framework.)

It helps you write applications that behave consistently, run in different environments (client, server, and native), and are easy to test. On top of that, it provides a great developer experience, such as live code editing combined with a time traveling debugger.

You can use Redux together with React, or with any other view library.
It is tiny (2kB, including dependencies).

Note: We are currently planning a rewrite of the Redux docs. Please take some time to fill out this survey on what content is most important in a docs site. Thanks!

Install

```
> npm i redux
```

Repository
github.com/reduxjs/redux

Homepage
redux.js.org

Weekly Downloads
8.841.731

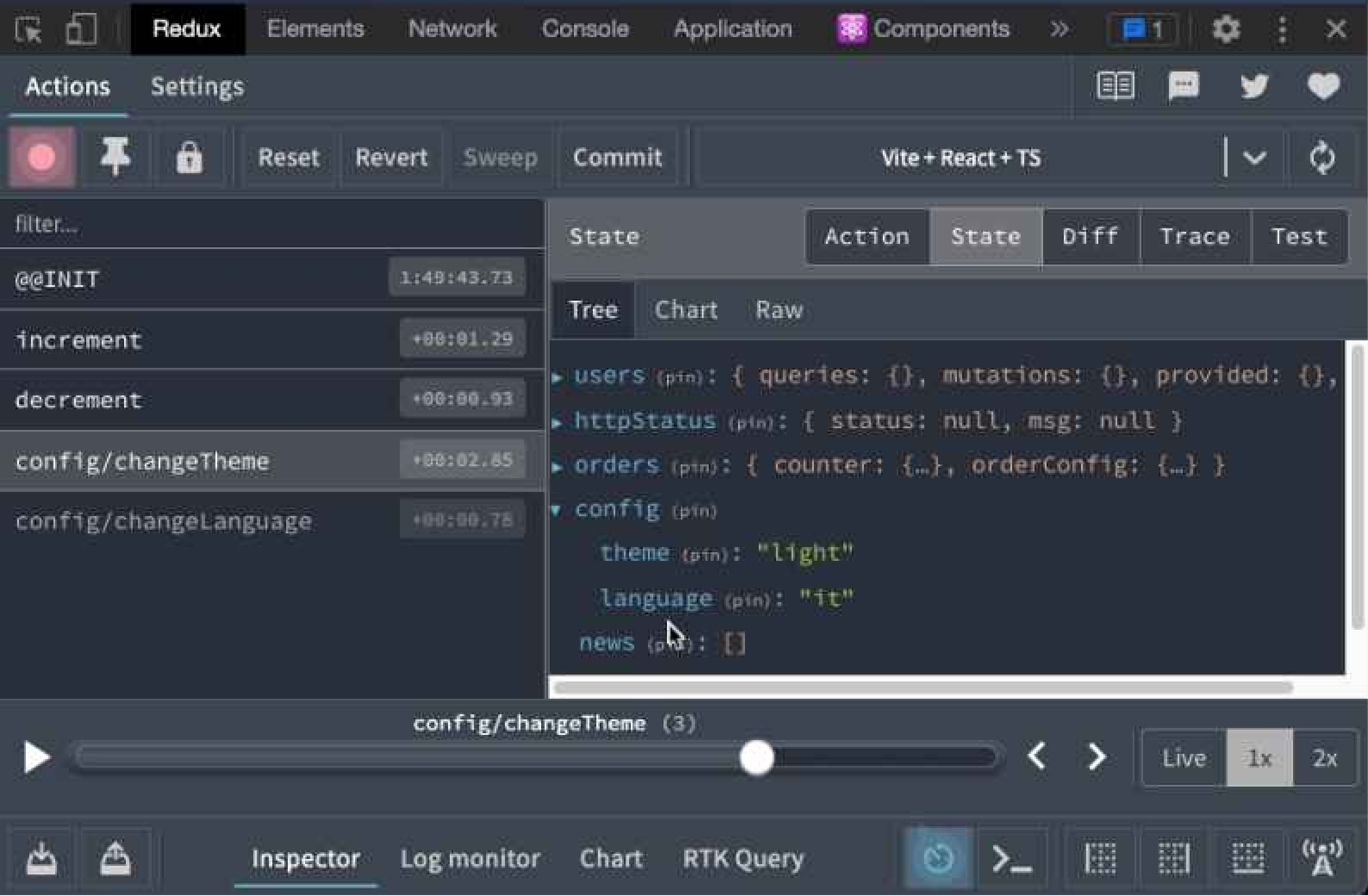
Version
4.2.1

License
MIT

Redux DevTools

One of the main benefits using Redux is **Redux DevTools**, a browser extension that allow you to:

- Easily **debug** your Redux app
- Always know what is happening in your app
- **Track actions** and how the **state changes**
- **time travel** debugging
- Export a **snapshot** and load it later to re-create a scenario
- display the state **chart**
- and more...



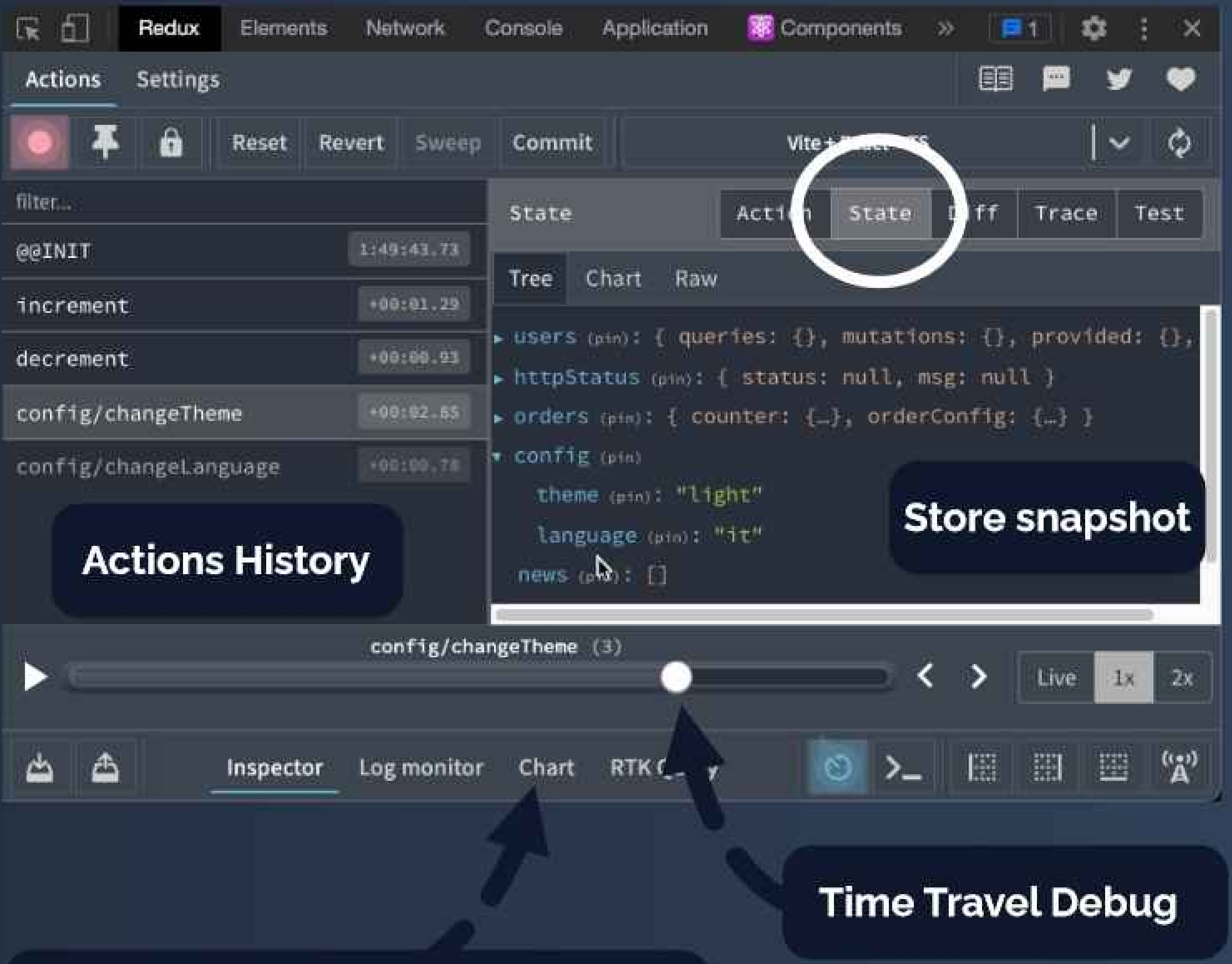
```
@@INIT                   1:49:43.73
increment                +00:01.29
decrement                +00:00.93
config/changeTheme       +00:02.85
config/changeLanguage    +00:00.78
```

State

```
▸ users (pin): { queries: {}, mutations: {}, provided: {},
▸ httpStatus (pin): { status: null, msg: null }
▸ orders (pin): { counter: {…}, orderConfig: {…} }
▾ config (pin)
    theme (pin): "light"
    language (pin): "it"
  news (pin): []
```

config/changeTheme (3)

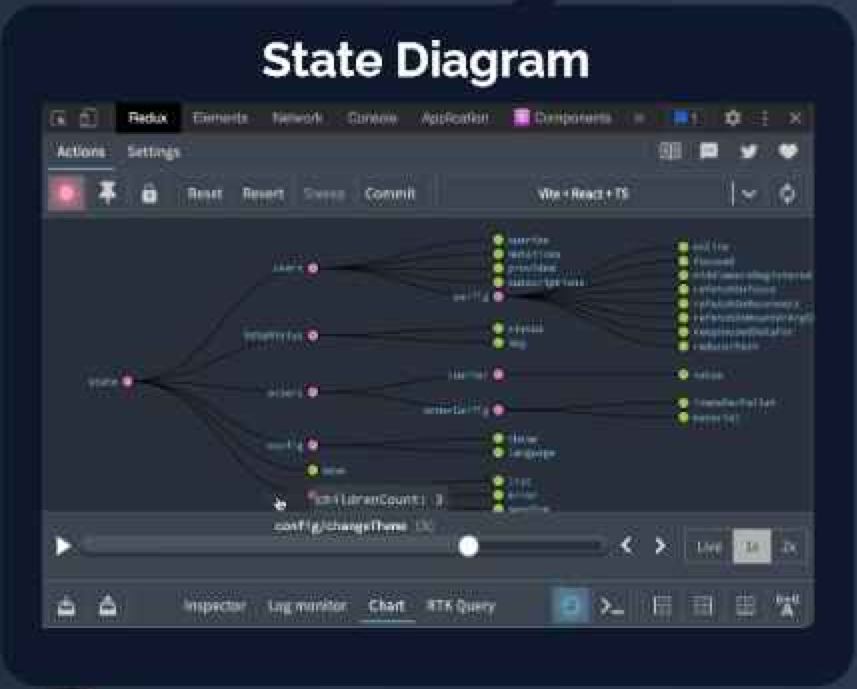
You can track everything that happens in your app



```
@@INIT                    1:49:43.73
increment                 +00:01.29
decrement                 +00:00.93
config/changeTheme        +00:02.65
config/changeLanguage     +00:00.78
```

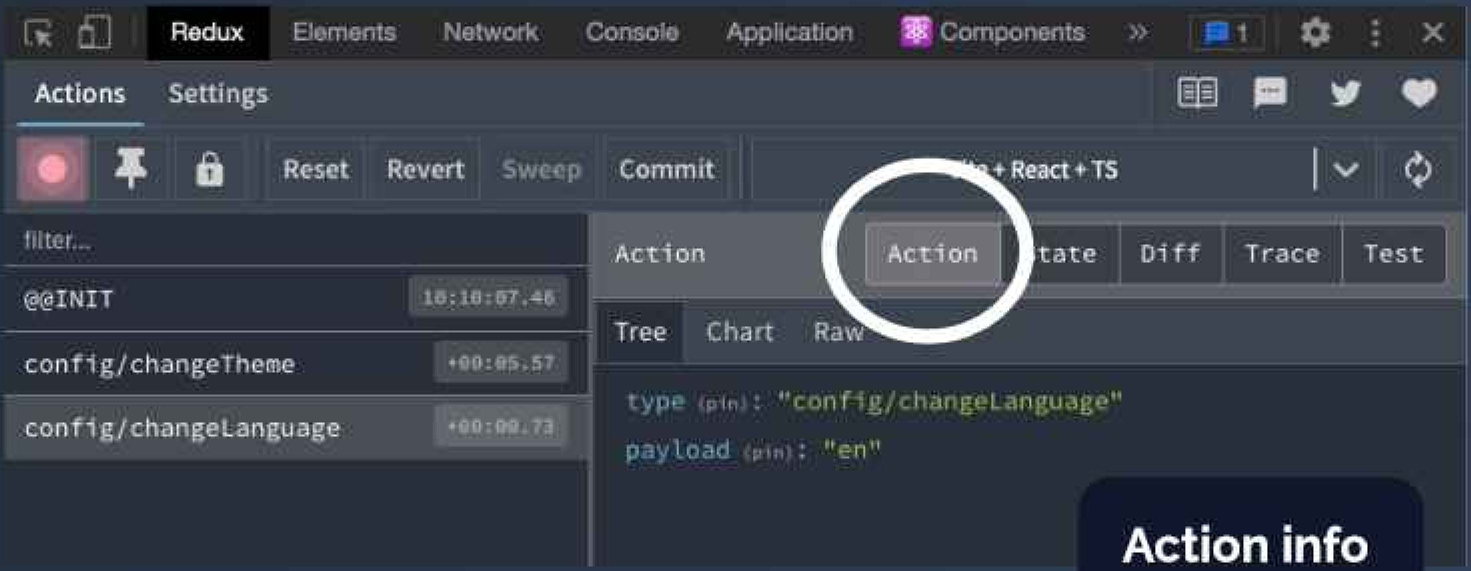
```
▸ users (pin): { queries: {}, mutations: {}, provided: {},
▸ httpStatus (pin): { status: null, msg: null }
▸ orders (pin): { counter: {…}, orderConfig: {…} }
▾ config (pin)
    theme (pin): "light"
    language (pin): "it"
  news (pin): []
```

config/changeTheme (3)



Redux DevTools

Track what actions you're dispatching (**type**)
and which parameters are you sending (**payload**)

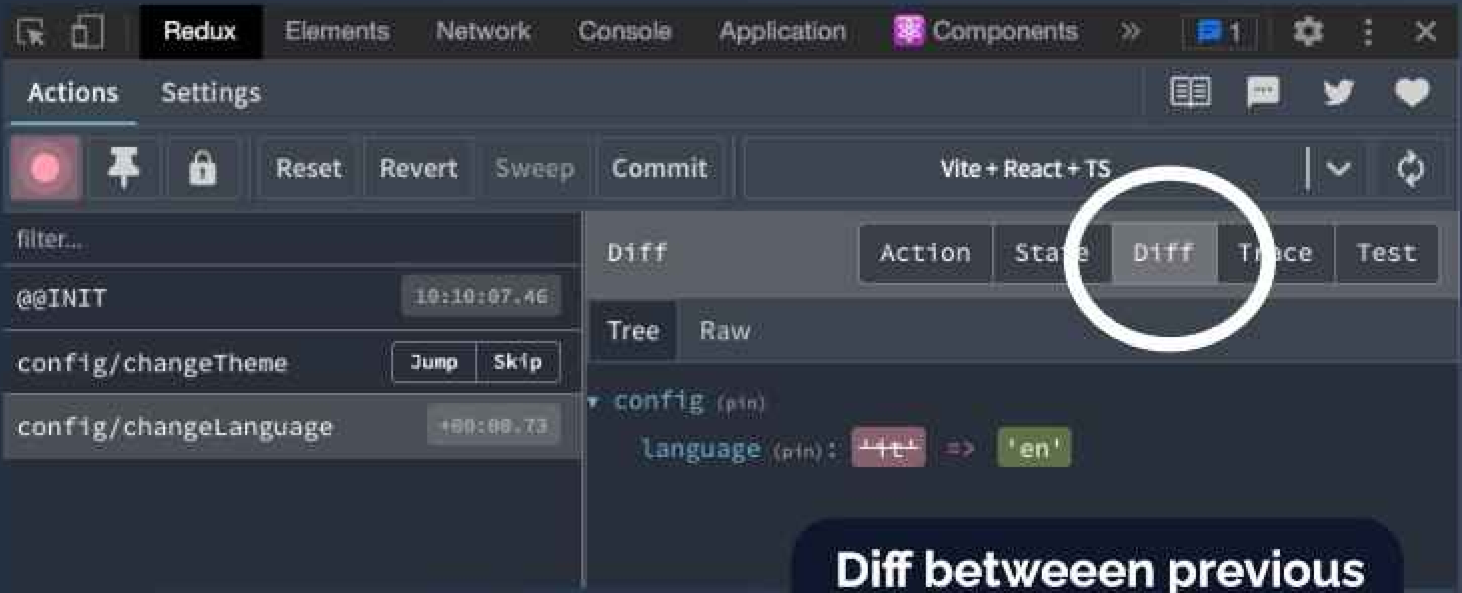


```
@@INIT                    10:10:07.46
config/changeTheme        +00:05.57
config/changeLanguage     +00:00.73
```

```
type (pin): "config/changeLanguage"
payload (pin): "en"
```

Action info

And how the state is changed
after an action has been dispatched



```
@@INIT                    10:10:07.46
config/changeTheme        Jump  Skip
config/changeLanguage     +00:00.73
```

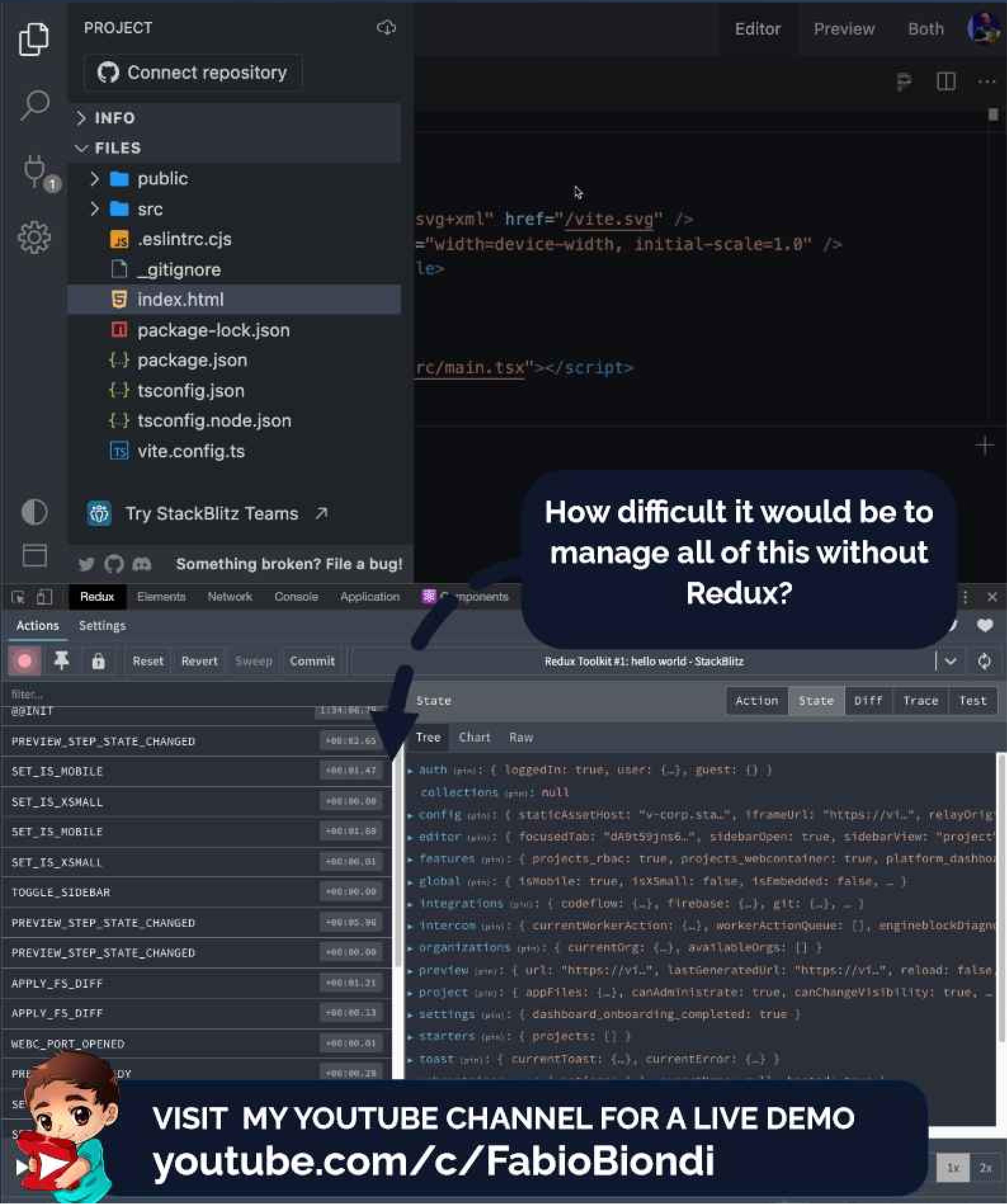
```
▾ config (pin)
    language (pin): ~~'it'~~ => 'en'
```

Diff betweeen previous and current state

fabiobiondi.dev

A real world example using Redux

StackBlitz



PROJECT

Connect repository

INFO

FILES

- public
- src
- .eslintrc.cjs
- .gitignore
- index.html
- package-lock.json
- package.json
- tsconfig.json
- tsconfig.node.json
- vite.config.ts

Try StackBlitz Teams

Something broken? File a bug!

Editor | Preview | Both

```
svg+xml" href="/vite.svg" />
="width=device-width, initial-scale=1.0" />
le>


rc/main.tsx"></script>
```

How difficult it would be to manage all of this without Redux?

```
@@INIT                          1:34:06.79
PREVIEW_STEP_STATE_CHANGED      +00:02.65
SET_IS_MOBILE                   +00:01.47
SET_IS_XSMALL                   +00:00.00
SET_IS_MOBILE                   +00:01.88
SET_IS_XSMALL                   +00:00.01
TOGGLE_SIDEBAR                  +00:00.00
PREVIEW_STEP_STATE_CHANGED      +00:05.96
PREVIEW_STEP_STATE_CHANGED      +00:00.00
APPLY_FS_DIFF                   +00:01.21
APPLY_FS_DIFF                   +00:00.13
WEBC_PORT_OPENED                +00:00.01
PRE                  DY         +00:00.28
SE
S
```

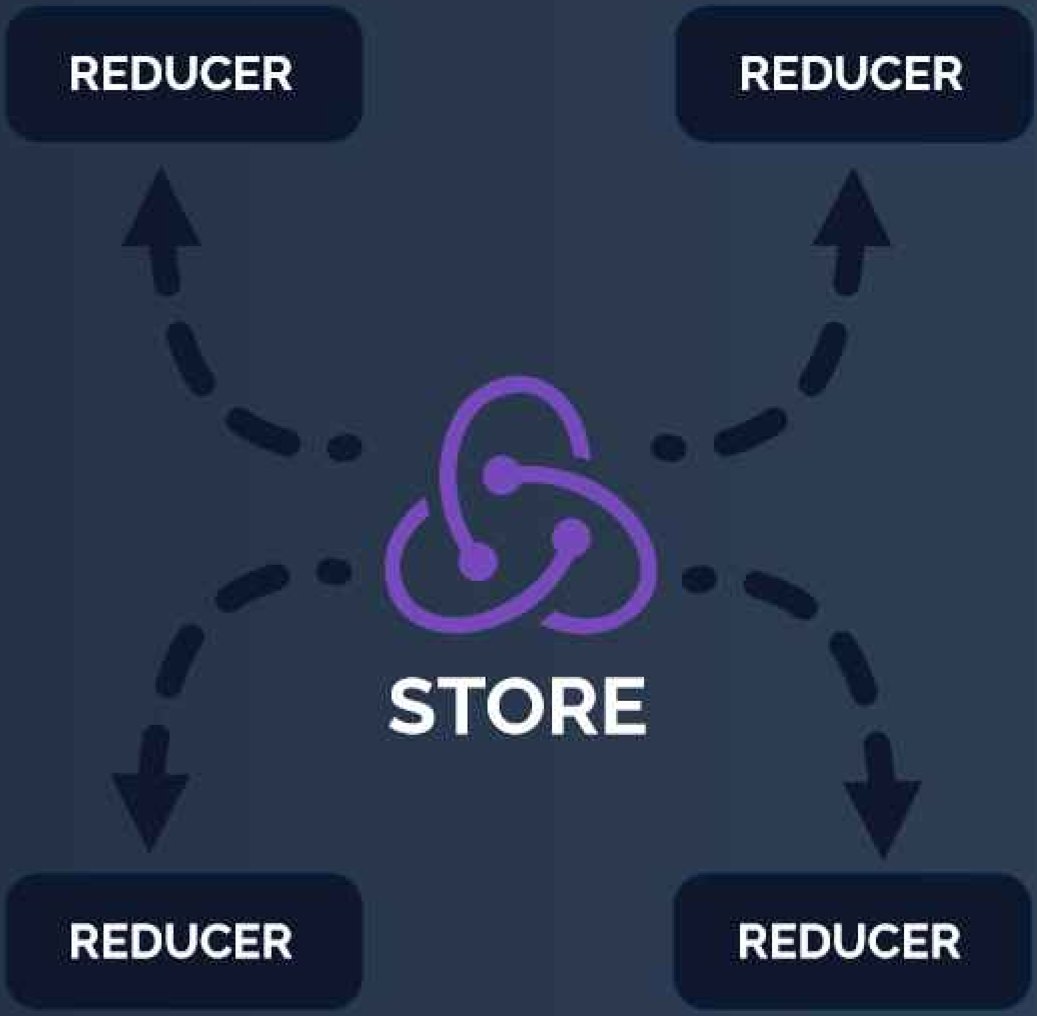
```
State
▸ auth (pin): { loggedIn: true, user: {…}, guest: {} }
  collections (pin): null
▸ config (pin): { staticAssetHost: "v-corp.sta…", iframeUrl: "https://vi…", relayOrig
▸ editor (pin): { focusedTab: "dA9t59jns6…", sidebarOpen: true, sidebarView: "project"
▸ features (pin): { projects_rbac: true, projects_webcontainer: true, platform_dashboa
▸ global (pin): { isMobile: true, isXSmall: false, isEmbedded: false, … }
▸ integrations (pin): { codeflow: {…}, firebase: {…}, git: {…}, … }
▸ intercom (pin): { currentWorkerAction: {…}, workerActionQueue: [], engineblockDiagno
▸ organizations (pin): { currentOrg: {…}, availableOrgs: [] }
▸ preview (pin): { url: "https://vi…", lastGeneratedUrl: "https://vi…", reload: false
▸ project (pin): { appFiles: {…}, canAdministrate: true, canChangeVisibility: true, …
▸ settings (pin): { dashboard_onboarding_completed: true }
▸ starters (pin): { projects: [] }
▸ toast (pin): { currentToast: {…}, currentError: {…} }
```

VISIT MY YOUTUBE CHANNEL FOR A LIVE DEMO

youtube.com/c/FabioBiondi

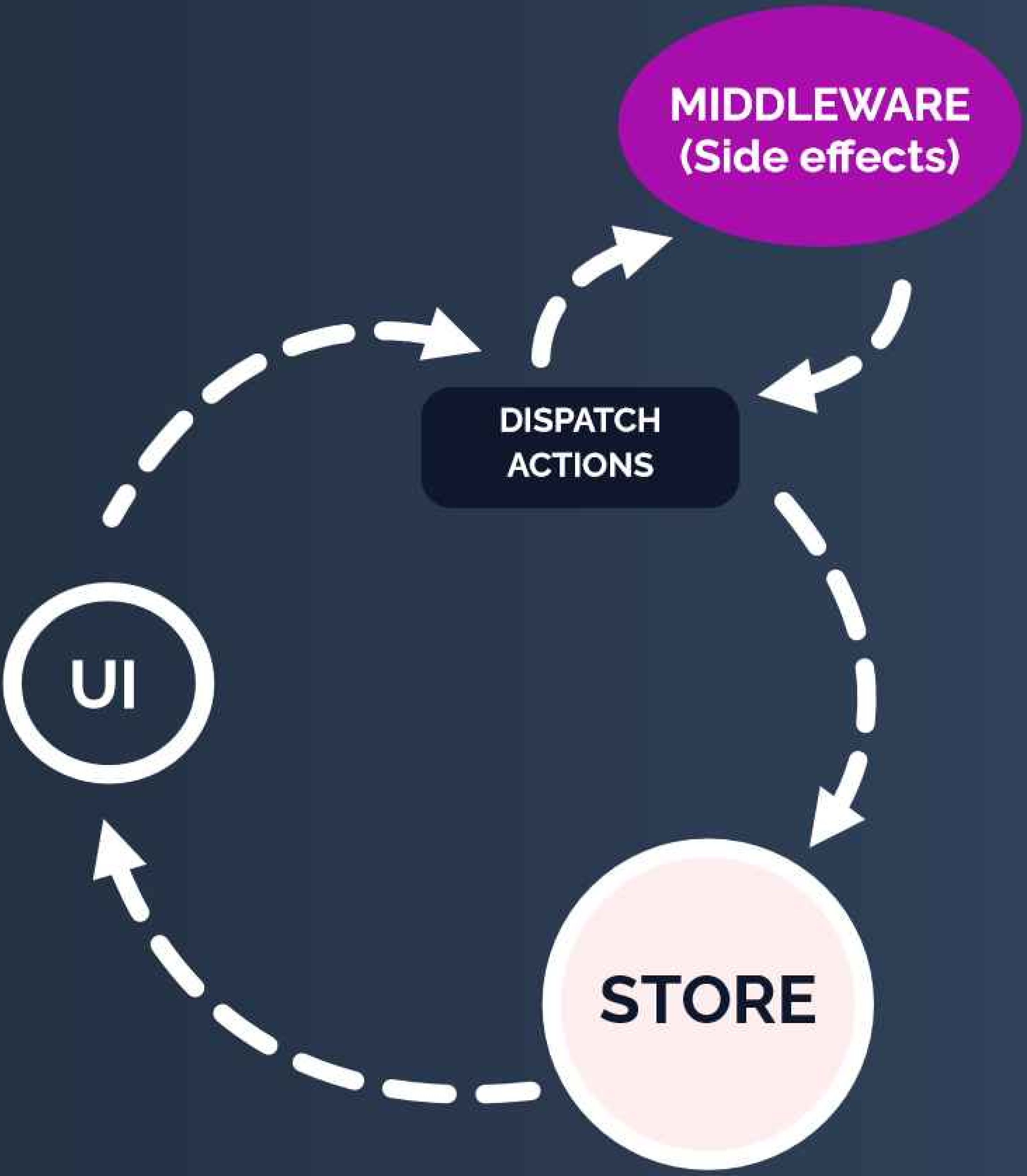
REDUX STORE

WHAT IS THE STORE?
The **store** is a global application state and it's composed by multiple reducers



WHAT IS A REDUCER?
It's a function that **initialize** and **mutate** a part of the store

REDUX FLOW



INSTALL DEPENDENCIES

Install Redux for React

```
npm i react-redux
```

it provides the **useSelector** and **useDispatch** hooks

Install Redux ToolKit (RTK)

```
npm i @reduxjs/toolkit
```

Include several utilities to simplify and enhance Redux

Configure the Store

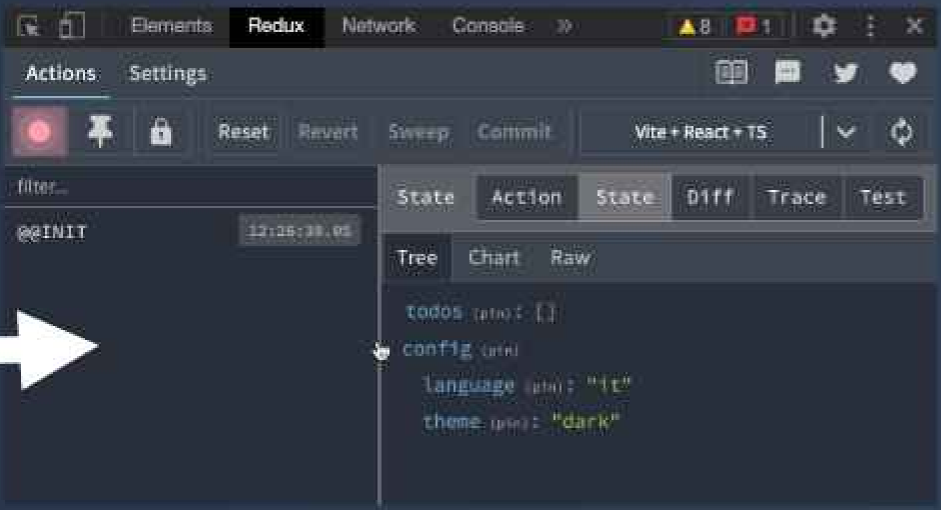
Store Creation

```
const rootReducer = combineReducers({
  todos: () => [],
  config: () => ({ theme: 'dark', lang: 'it'})
});


export const store = configureStore({
  reducer: rootReducer,
});


export type RootState = ReturnType<typeof rootReducer>;
```

Store Type



```
@@INIT    12:26:38.05
```

```
todos (pin): []
config (pin)
  language (pin): "it"
  theme (pin): "dark"
```

Wrap your app with Provider

```
<Provider store={store}>
    <App />
</Provider>
```

fabiobiondi.dev

ACTIONS

Payload Type

Action Type

counter.actions.ts

```ts
export const increment = createAction<number>('increment')
export const decrement = createAction<number>('decrement')
```

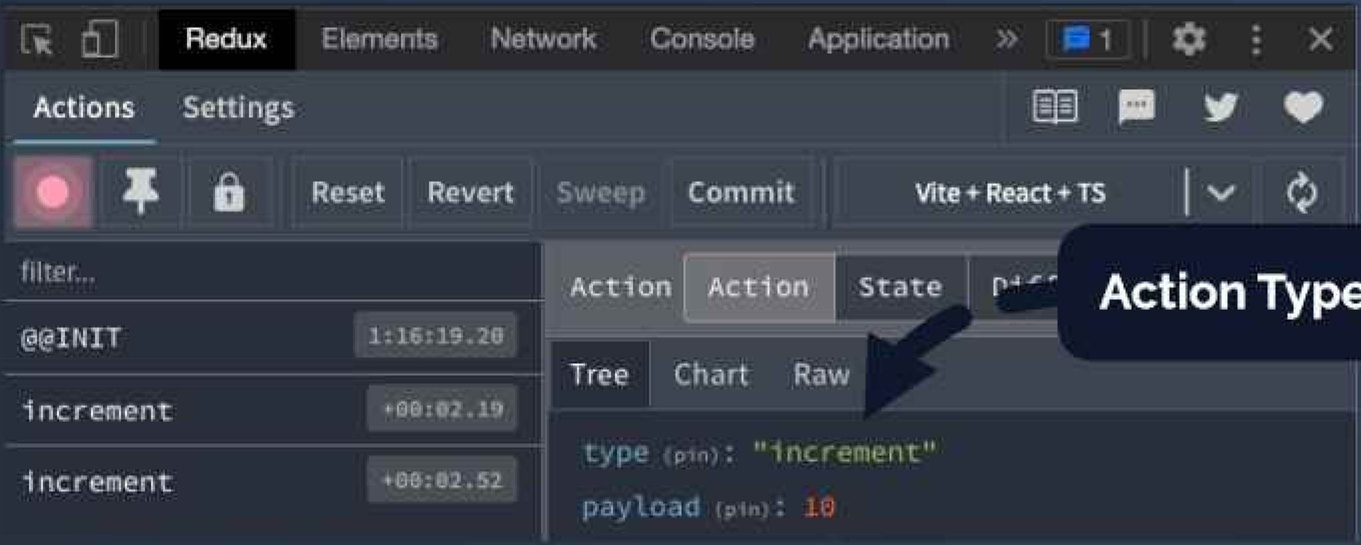
MyComponent.tsx

Dispatch Action

```tsx
export const Component = () => {
  const dispatch = useDispatch();

  return <div>
    <button onClick={() => dispatch(increment(10))}>+</button>
    <button onClick={() => dispatch(decrement(5))}>-</button>
  </div>
};
```

REDUX DEV TOOLS



```
@@INIT        1:16:19.20
increment     +00:02.19
increment     +00:02.52
```

```
type (pin): "increment"
payload (pin): 10
```

Action Type

Payload

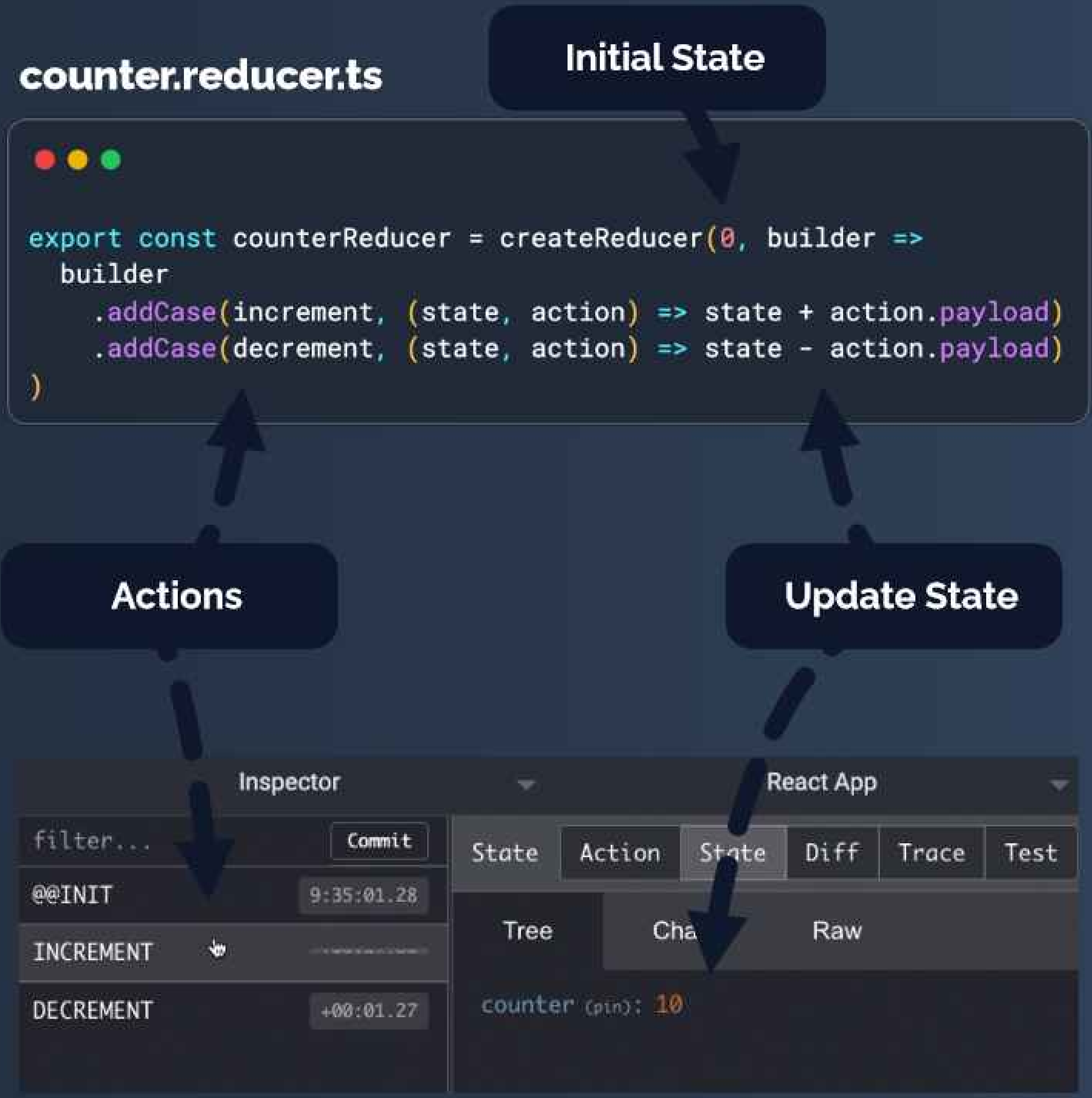
fabiobiondi.dev

REDUCER

WHAT IS A REDUCER?

It's a function that **initialize** and **mutate** a part of the store in according to the dispatched actions

counter.reducer.ts



```
export const counterReducer = createReducer(0, builder =>
  builder
    .addCase(increment, (state, action) => state + action.payload)
    .addCase(decrement, (state, action) => state - action.payload)
)
```

```
@@INIT        9:35:01.28
INCREMENT
DECREMENT     +00:01.27
```

```
counter (pin): 10
```

combineReducers

combines multiple reducers in one store

ADD REDUCERS to store

```
const rootReducer = combineReducers({
  counter: counterReducer,
  config: configReducer,
  todos: todoReducer,
  auth: authReducer,
  // ...
});
```

WHAT IS THE STORE?

The **store** is a global application state and it's composed by multiple reducers

createSlice

An utility to simplify the creation of reducers and actions in one place

(So it's an alternative to the previous approach)

```
const initialState: ConfigState = { language: 'it', theme: 'dark' };

export const configStore = createSlice({
  name: 'config',
  initialState,
  reducers: {
    changeTheme(state, action: PayloadAction<Theme>) {
      state.theme = action.payload;
    },
    changeLanguage(state, action: PayloadAction<Language>) {
      state.language = action.payload;
    },
  },
});

export const { changeLanguage, changeTheme } = configStore.actions;
```

ACTIONS

STATE UPDATE

EXPORT ACTIONS

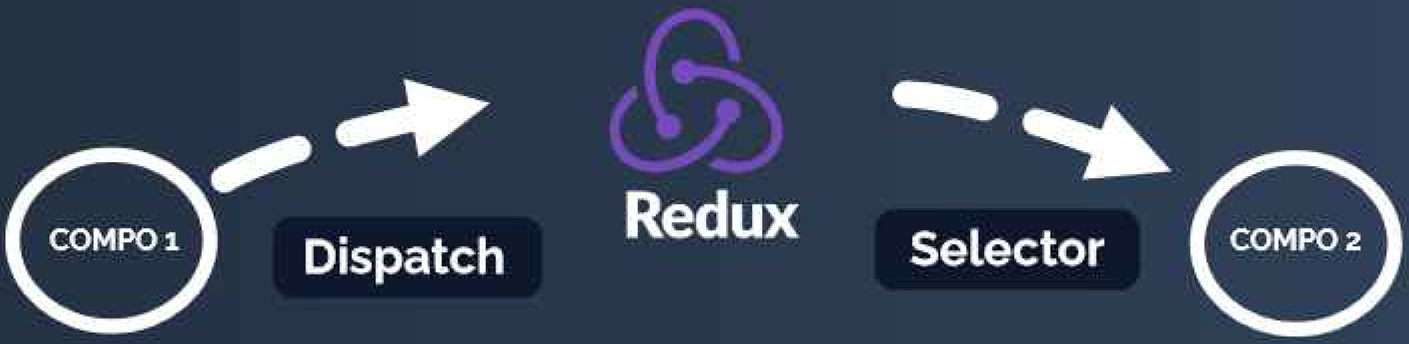
HINT: IMMER JS

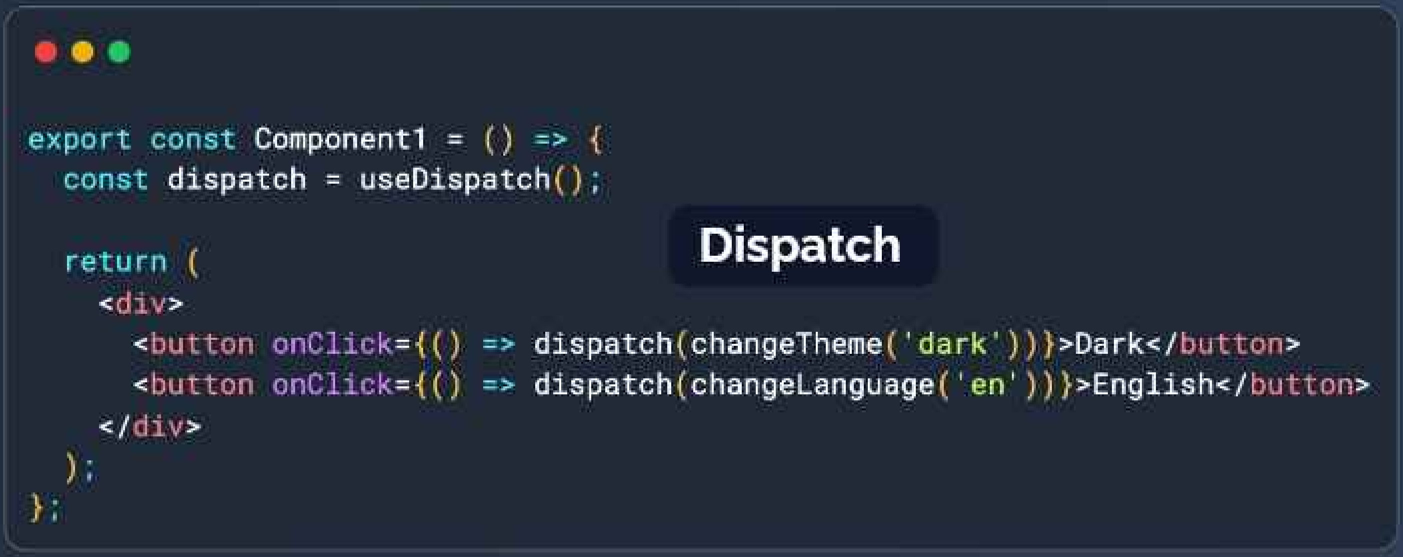
Redux does not allow to directly mutate the state
but it's allowed in Redux Toolkit since it includes immerJS

```
const rootReducer = combineReducers({
  config: configStore.reducer
});
```

Add slice to Store

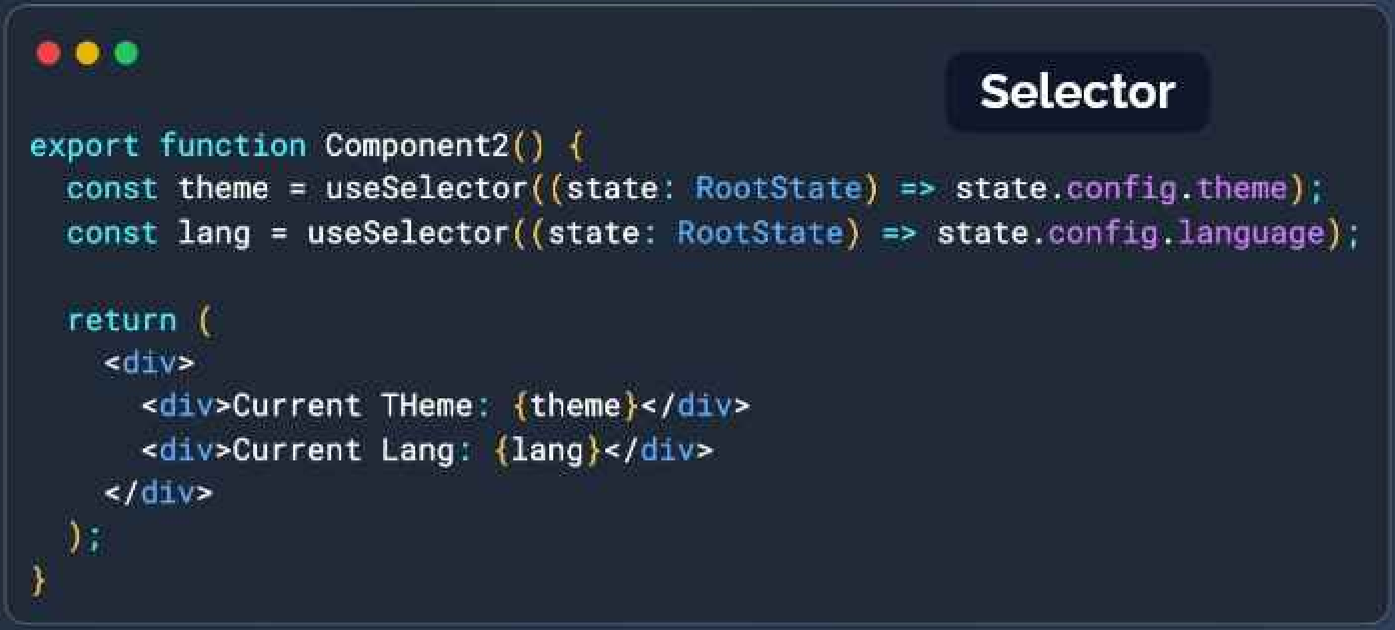
USAGE





```
export const Component1 = () => {
  const dispatch = useDispatch();

  return (
    <div>
      <button onClick={() => dispatch(changeTheme('dark'))}>Dark</button>
      <button onClick={() => dispatch(changeLanguage('en'))}>English</button>
    </div>
  );
};
```



```
export function Component2() {
  const theme = useSelector((state: RootState) => state.config.theme);
  const lang = useSelector((state: RootState) => state.config.language);

  return (
    <div>
      <div>Current THeme: {theme}</div>
      <div>Current Lang: {lang}</div>
    </div>
  );
}
```

But there is much more...

FOLLOW ME & LIKE
to get more info about
Front-End Development