



A quick guide about **Redux** & **Redux Toolkit(RTK)**

In React / TypeScript



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

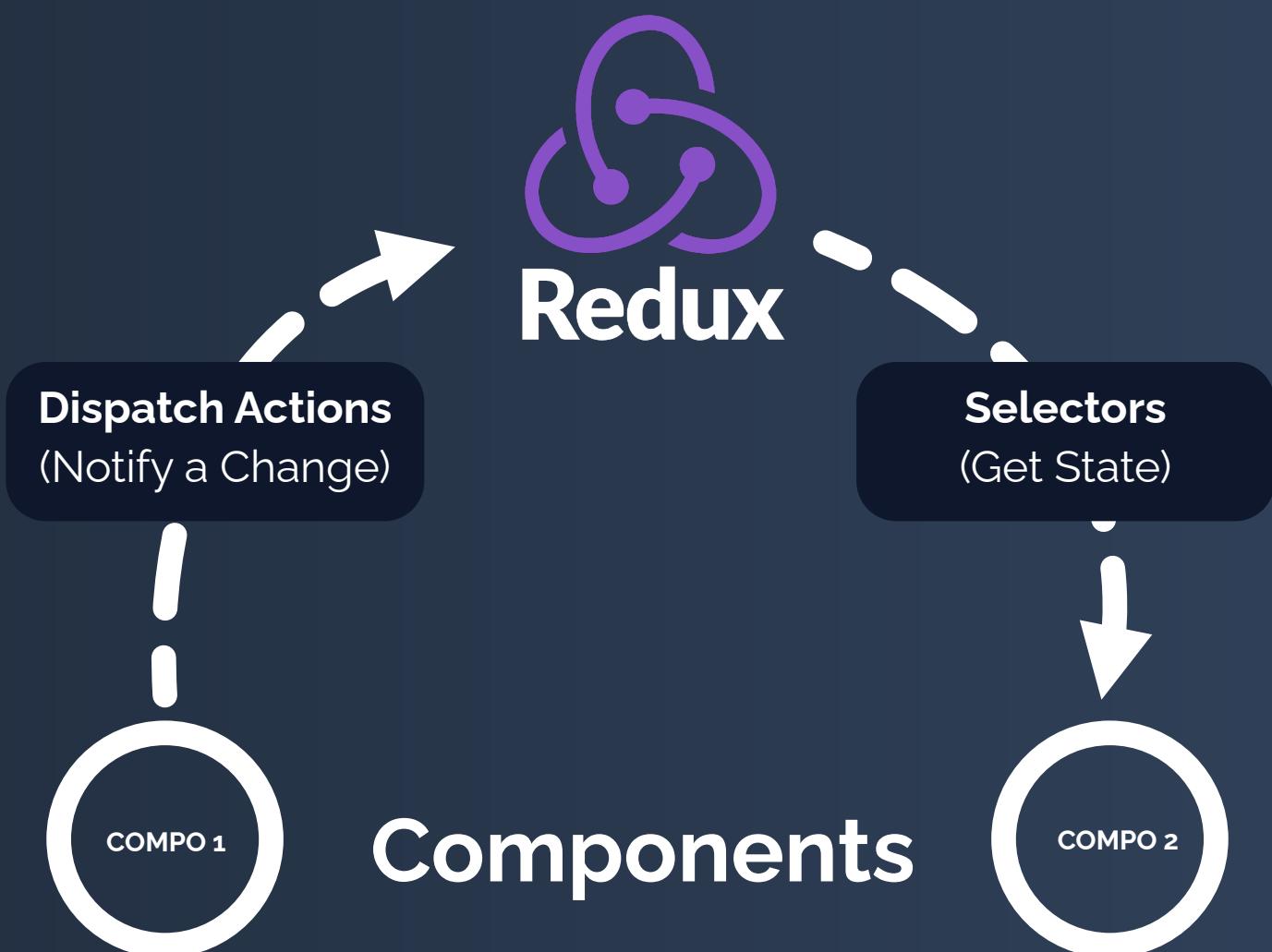
Time Travel Debugging: Redux's **powerful DevTools** extension allows for time travel debugging. You can **replay actions** and **inspect** the **state** at any point in time, making it a breeze to track down bugs and understand how your application's state evolves



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 common 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
```

PROJECT
FOLDERS



Who uses Redux?

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



8.8M download / week

npm

Search packages

Search

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



fabiobiondi.dev

5 / 21

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...

The screenshot shows the Redux DevTools interface integrated into a browser's developer tools. The top navigation bar includes tabs for Redux, Elements, Network, Console, Application, Components, and a pinned tab for 'Vite + React + TS'. Below the tabs are buttons for Actions (highlighted), Settings, and various tools like Reset, Revert, Sweep, and Commit. On the left, a sidebar lists actions with their timestamps: @@INIT (1:49:43.73), increment (+00:01.29), decrement (+00:00.93), config/changeTheme (+00:02.85), and config/changeLanguage (+00:00.78). The main panel displays the state tree. It has tabs for State, Action, State, Diff, Trace, and Test, with State selected. Under State, there are tabs for Tree, Chart, and Raw, with Tree selected. The state structure is shown as a hierarchical tree with expandable nodes: users, httpStatus, orders, and config. The config node is expanded, showing theme (light), language (it), and news (empty array). At the bottom, a timeline shows a single action: config/changeTheme (3), with a slider and playback controls for Live, 1x, and 2x.

You can track everything that happens in your app

The screenshot shows the React DevTools interface with the Redux tab selected. At the top, there are buttons for Actions and Settings. Below the Actions tab, there are icons for Reset, Revert, Sweep, and Commit. A search bar labeled "filter..." is present. A list of actions is shown with their names and timestamps:

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

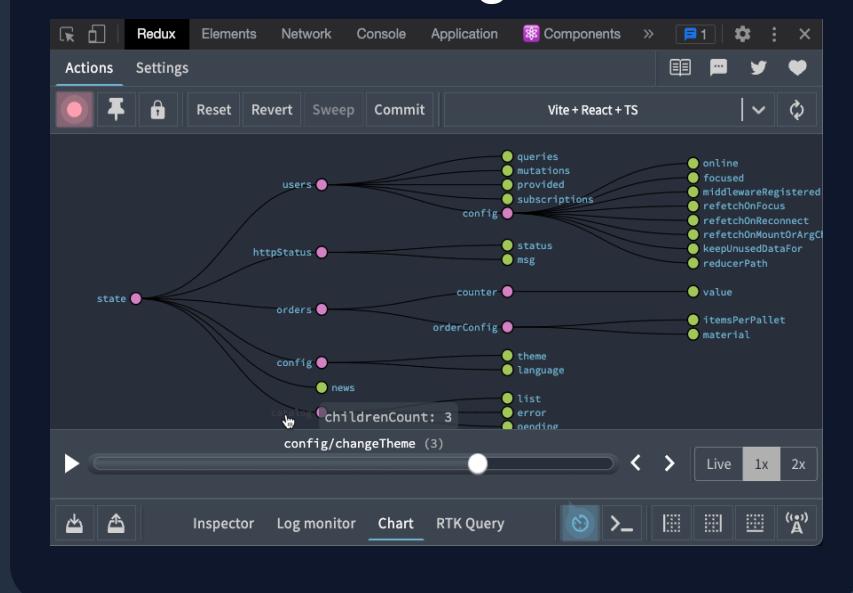
A large button labeled "Actions History" is overlaid on the left side of the actions list.

In the main pane, a "Vite + React + TS" project is selected. The "State" tab is active, showing a tree view of the application's state structure. A white circle highlights the "State" tab in the navigation bar. The state tree includes nodes for users, httpStatus, orders, config, news, and counter. A "Store snapshot" button is located in the bottom right corner of the state pane.

Below the state pane, a timeline slider is set to "config/changeTheme (3)". To the right of the slider, there are buttons for "Live", "1x", and "2x". The "RTK Query" tab is also visible in the bottom navigation bar.

Time Travel Debug

State Diagram



Redux DevTools

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

A screenshot of the Redux DevTools extension for Chrome. The interface has a dark theme. At the top, there's a navigation bar with tabs: Redux, Elements, Network, Console, Application, Components, and a few more on the right. Below the navigation bar is a toolbar with icons for Actions (a pink circle), Settings, Reset, Revert, Sweep, and Commit. A large list of actions is displayed on the left, each with a timestamp. On the right, there's a detailed view of an action, with a callout bubble labeled "Action info". The "Action" tab is highlighted with a white circle. The detailed view shows the action type as "config/changeLanguage" and the payload as "en".

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

A screenshot of the Redux DevTools extension for Chrome, similar to the previous one but with a different tab selected. The "Diff" tab is highlighted with a white circle. The detailed view on the right shows the difference between the previous state and the current state for the "config" key. It highlights the change in the "language" field from "it" to "en". A callout bubble labeled "Diff between previous and current state" points to this detail.



A real world example using Redux



PROJECT Connect repository Editor Preview Both

> 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!

Redux Elements Network Console Application Components

Actions Settings

filter... @@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 +00:01.60 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 WEB_C_PORT_OPENED +00:00.01 PREPARE_READY +00:00.20

Redux Toolkit #1: hello world - StackBlitz

State Action State Diff Trace Test

Tree Chart Raw

- 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: [], engineblockDiagn...
- 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: {} }

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

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

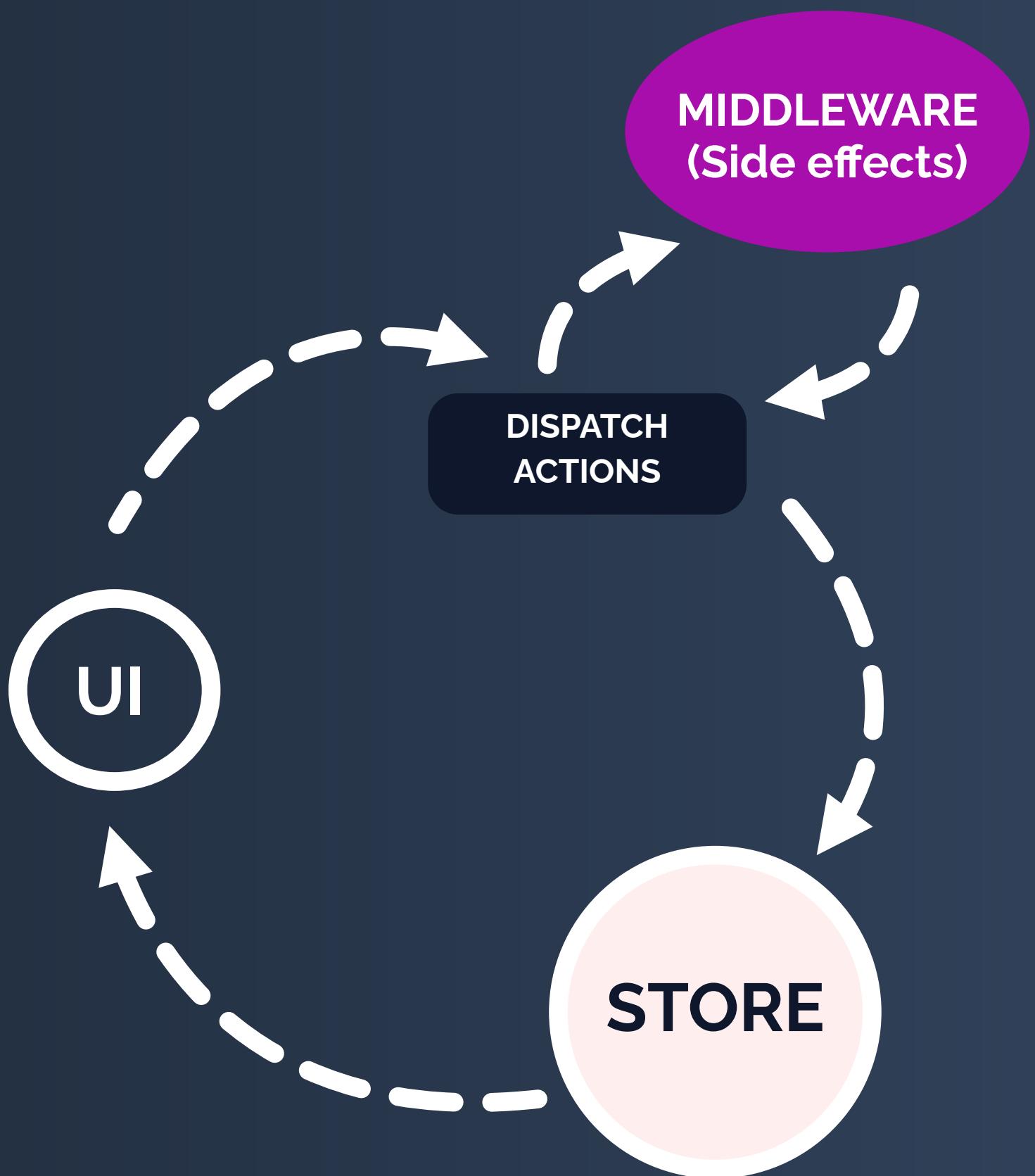


WHAT IS A REDUCER?

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

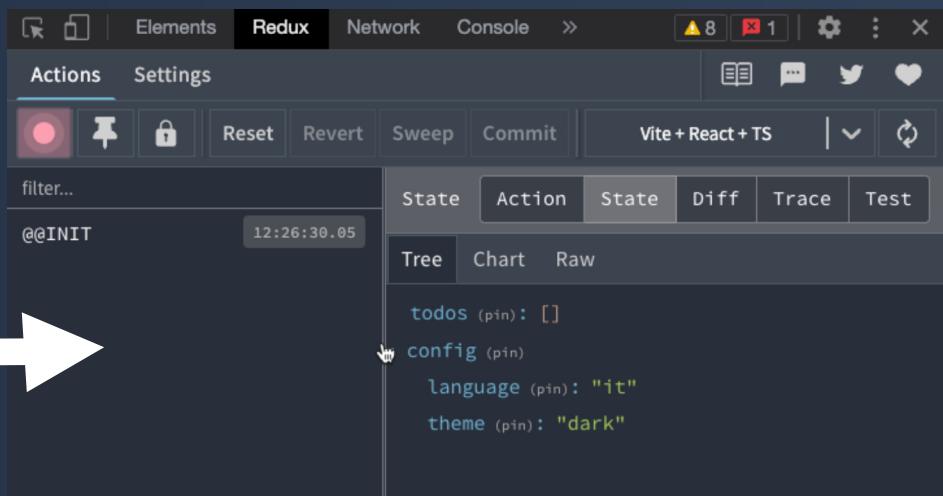
Store Creation

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

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

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

Store Type



Wrap your app with Provider

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



ACTIONS

counter.actions.ts

Payload Type

Action Type



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

MyComponent.tsx



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

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

Dispatch Action

REDUX DEV TOOLS

Action Type

The screenshot shows the Redux Dev Tools interface. At the top, there's a toolbar with various icons and tabs like 'Redux', 'Elements', 'Network', 'Console', and 'Application'. Below the toolbar, there are two tabs: 'Actions' (selected) and 'Settings'. On the left, a list of actions is shown with columns for 'filter...', 'Action', 'State', and 'Diff'. Actions listed include '@@INIT' at 1:16:19.20, 'increment' at +00:02.19, and another 'increment' at +00:02.52. On the right, a detailed view of an 'increment' action is expanded, showing its type ('increment') and payload ('10'). Below this, there are tabs for 'Tree', 'Chart', and 'Raw'.

filter...	Action	State	Diff
@@INIT	1:16:19.20		
increment	+00:02.19		
increment	+00:02.52		

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

Payload



REDUCER



WHAT IS A REDUCER?

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

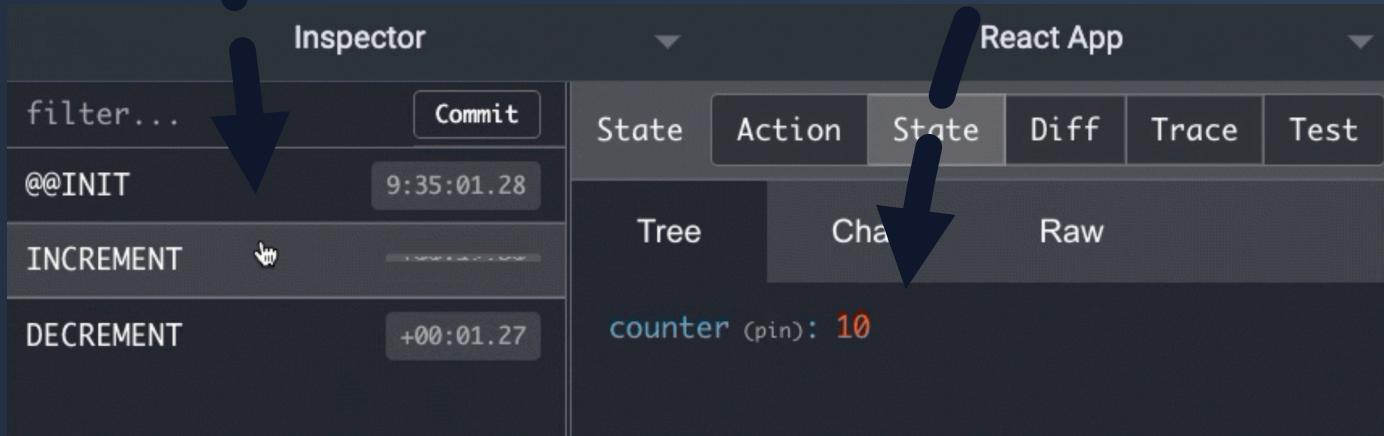
counter.reducer.ts

Initial State

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

Actions

Update State



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>
  );
};
```

Dispatch



Selector

```
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



linkedin.com/in/fabiobiondi

19/21



▶ VIDEO CORSO



REACT PRO

Creare Real World Applications

React 18, TypeScript, Zustand, Tailwind, Vite

160+
video

8+
ore di
lezione

12
capitoli



ESTRATTO ARGOMENTI

- Creare shop con area riservata
- React & **TypeScript**
- Utilizzare **pochissime dipendenze**
- Creazione layout con **Tailwind**
- Custom Components e Custom hooks
- Local State con **useReducer**
- State Management con **Zustand**
- **Autenticazione** con JWT
- Good Practices, Clean Coding & Tips



DISPONIBILE SU
fabiobiondi.dev

ADS



fabiobiondi.dev



youtube.com/c/FabioBiondi

21/21