ReactJS Training Course (Session 4)

TUAN MAI CHUNG / NashTech

Sep/2020

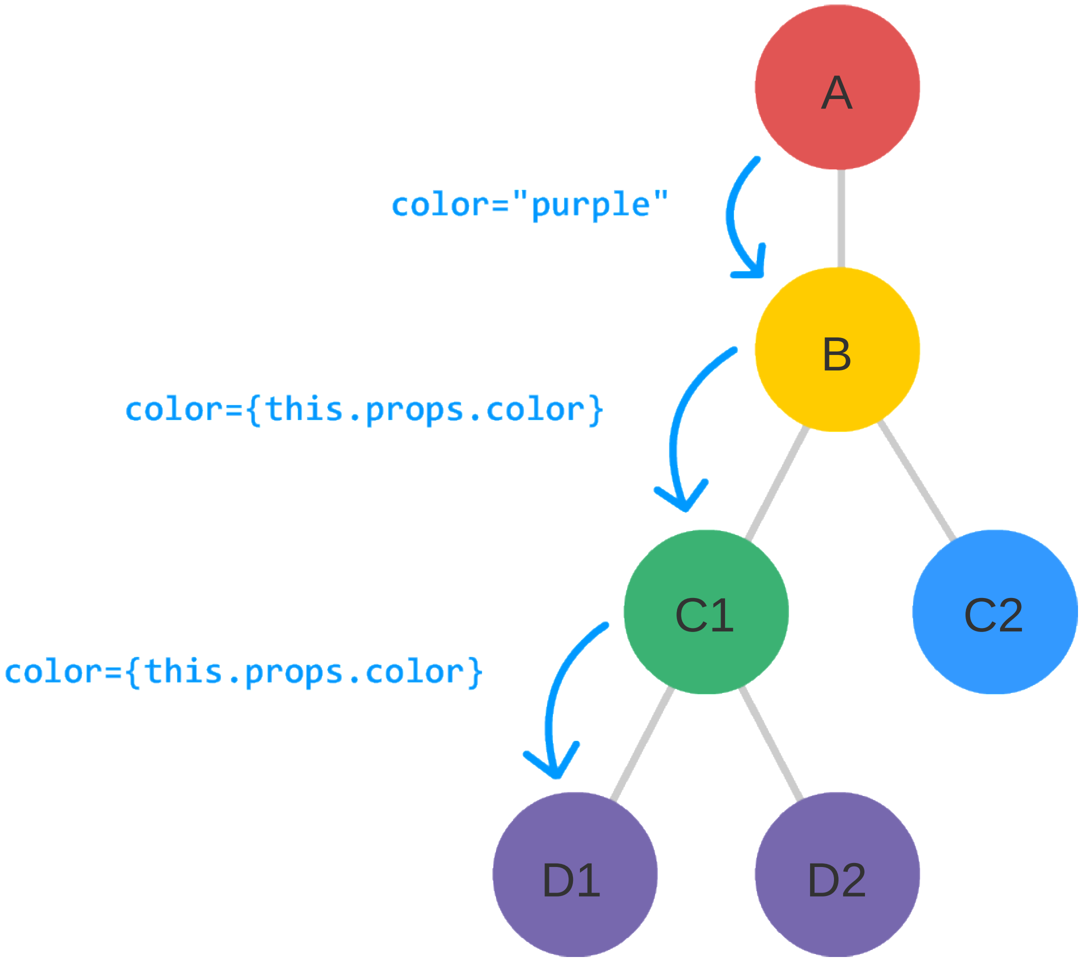
Agenda

1. CONTEXT - PROVIDER
2. UNDERSTAND REDUX
3. REACT DEVELOPER TOOL – REDUX TOOL
4. REDUX SAGA & REDUX THUNK

Goals

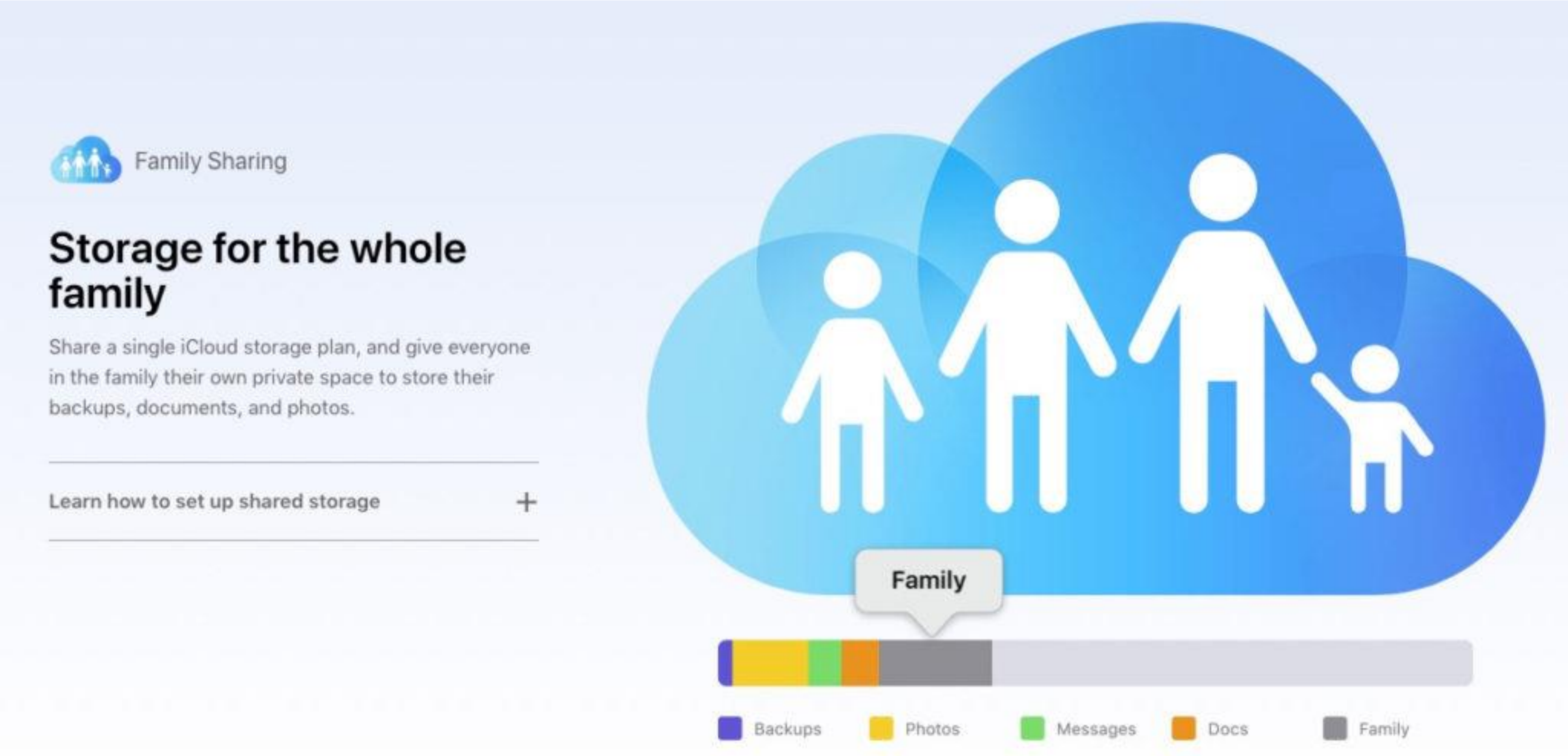
- Understand Redux and how to use Redux in React project.
- Understand how to debugger in React. Know how to use React Developer tool and Redux tool.

Traditional data transfer method



Issues?

Solution



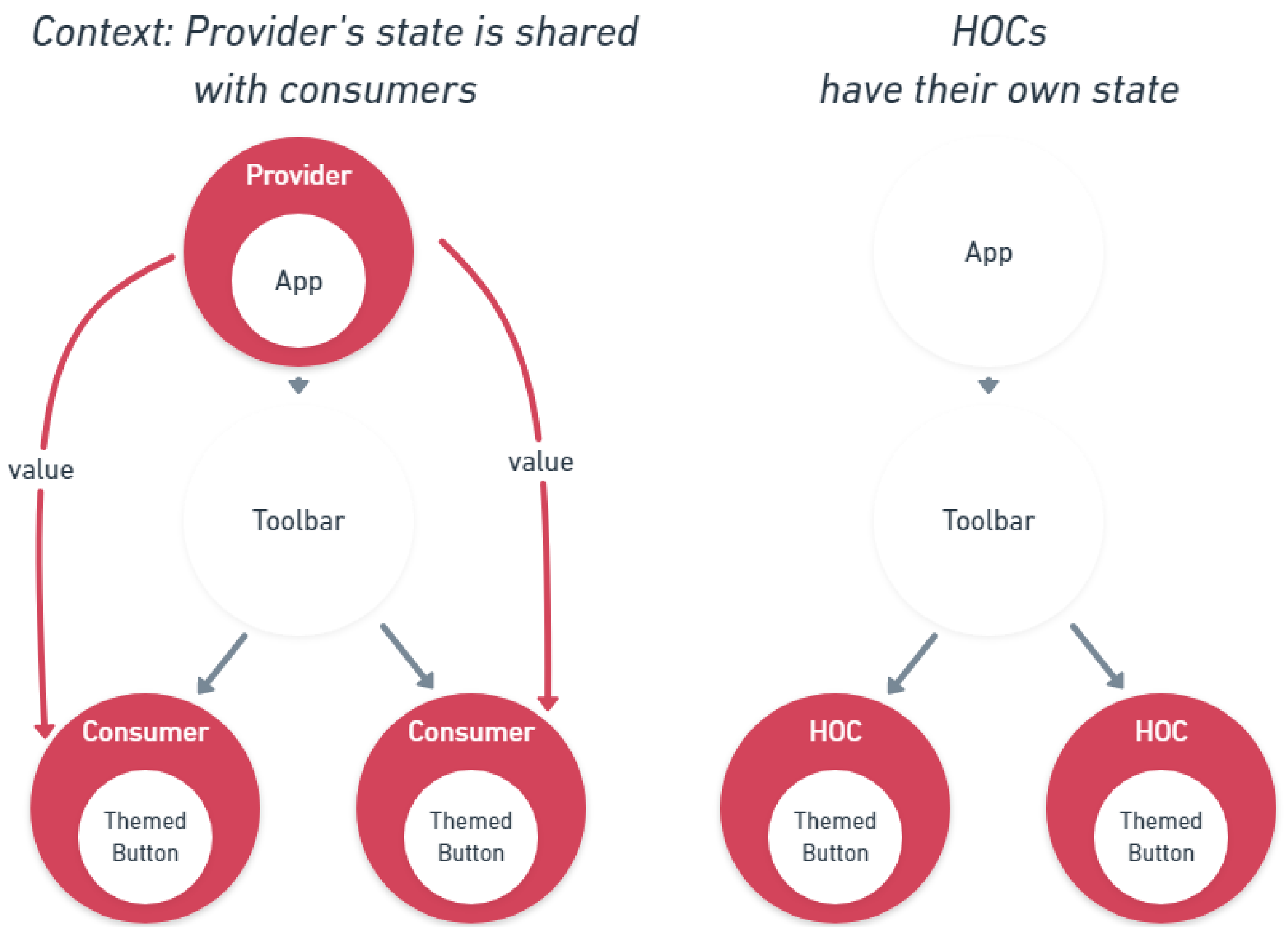
Family Sharing

Storage for the whole family

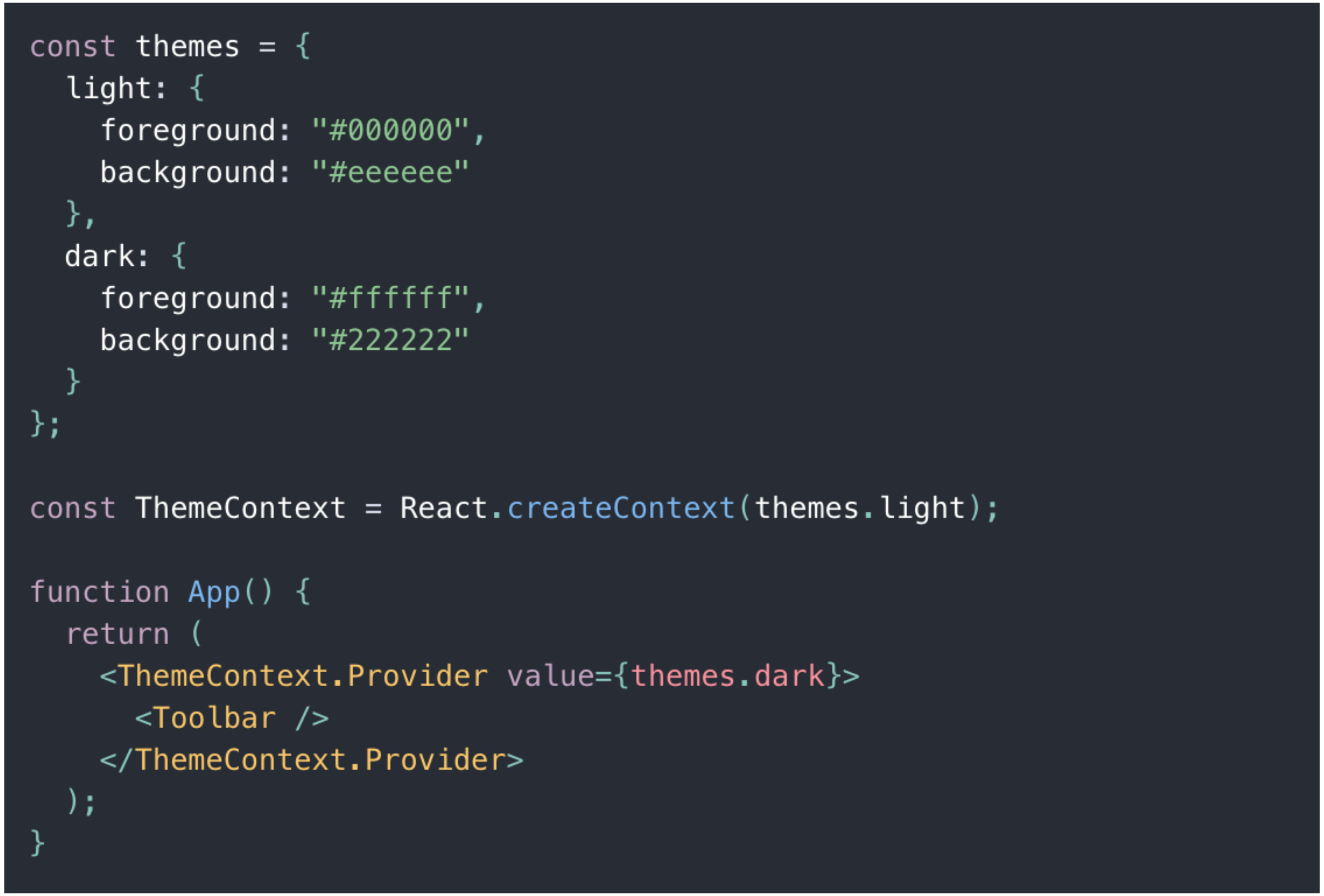
Share a single iCloud storage plan, and give everyone in the family their own private space to store their backups, documents, and photos.

Learn how to set up shared storage +

Context - Provider

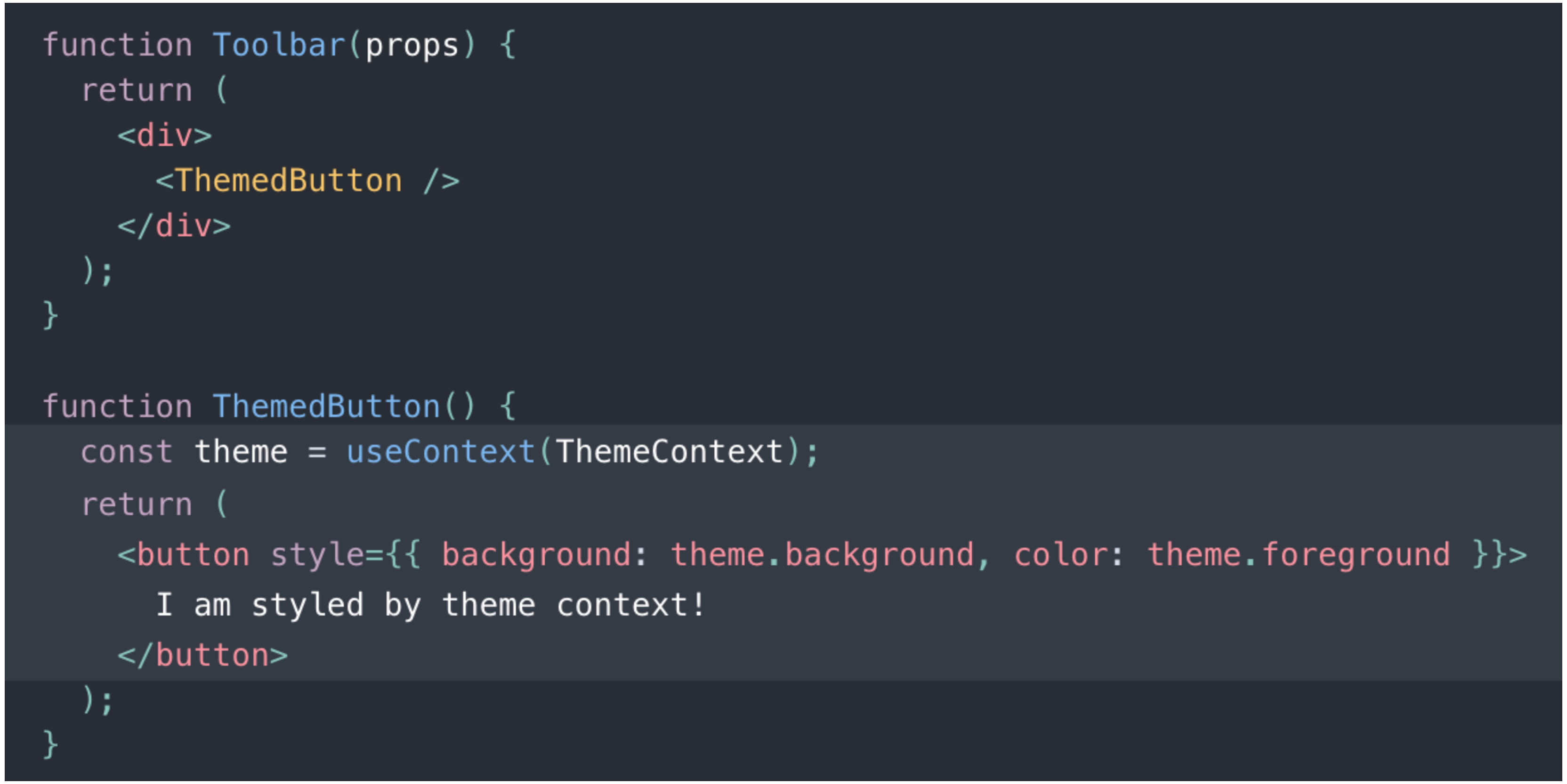


createContext()



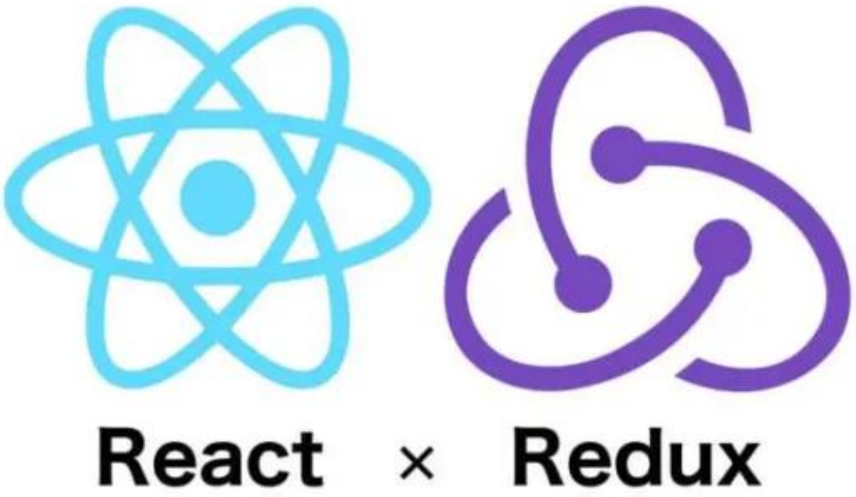
```
const themes = {
  light: {
    foreground: "#000000",
    background: "#eeeeee"
  },
  dark: {
    foreground: "#ffffff",
    background: "#222222"
  }
};

const ThemeContext = React.createContext(themes.light);

function App() {
  return (
    <ThemeContext.Provider value={themes.dark}>
      <Toolbar />
    </ThemeContext.Provider>
  );
}
```

useContext()



```
function Toolbar(props) {
  return (
    <div>
      <ThemedButton />
    </div>
  );
}

function ThemedButton() {
  const theme = useContext(ThemeContext);
  return (
    <button style={{ background: theme.background, color: theme.foreground }}>
      I am styled by theme context!
    </button>
  );
}
```

Redux



React Redux

- Redux is a state container for JavaScript apps, often called a Redux store.
- It can be used with other frameworks such as Angular, Ember or Vue.js.
- Redux consists of three main components:
  - Store
  - Actions
  - Reducer

React Redux- 3 Principles

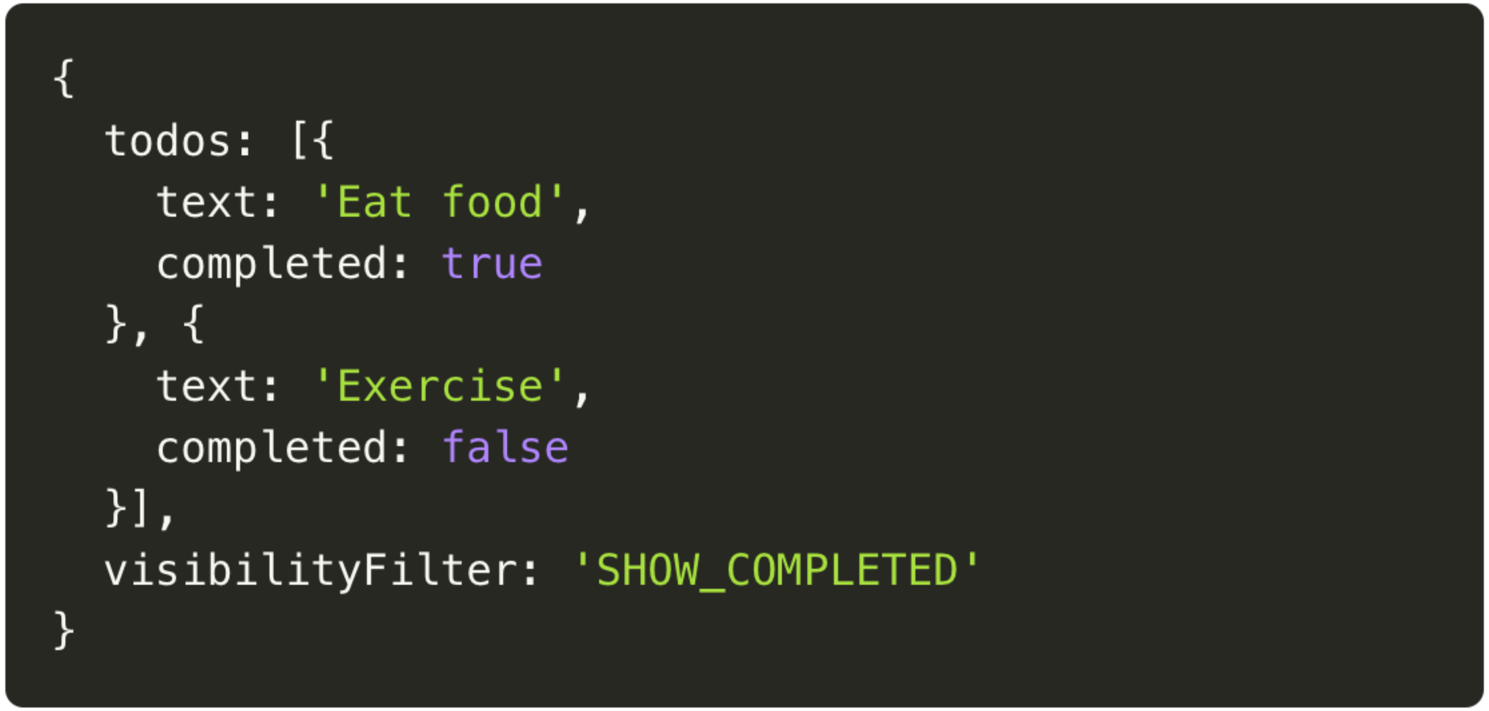
- Single source of truth
- State is **read-only**
- Changes are made with pure functions

React Redux- Installation

- With NPM:
  - **npm install redux react-redux**
- With Yarn
  - **yarn add redux react-redux**

React Redux - Store

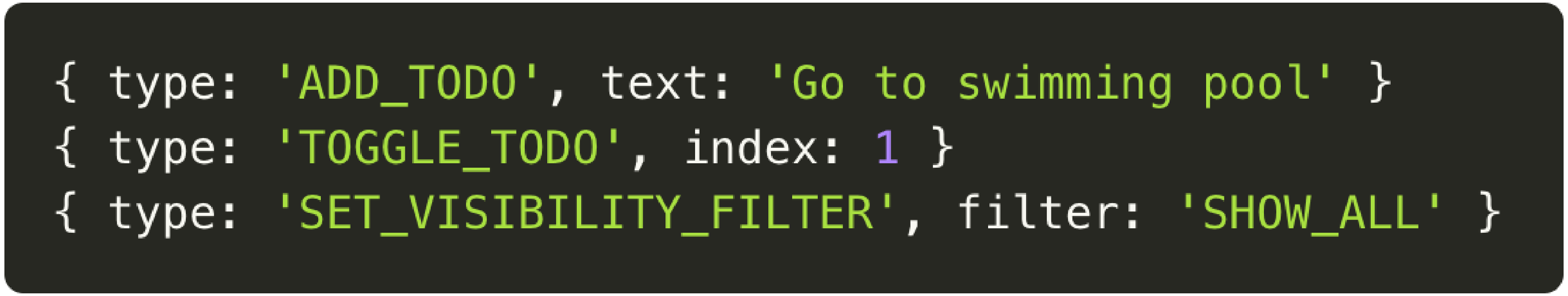
- Your app’s state is described as a plain object



```
{
  todos: [{
    text: 'Eat food',
    completed: true
  }, {
    text: 'Exercise',
    completed: false
  }],
  visibilityFilter: 'SHOW_COMPLETED'
}
```

React Redux - Action

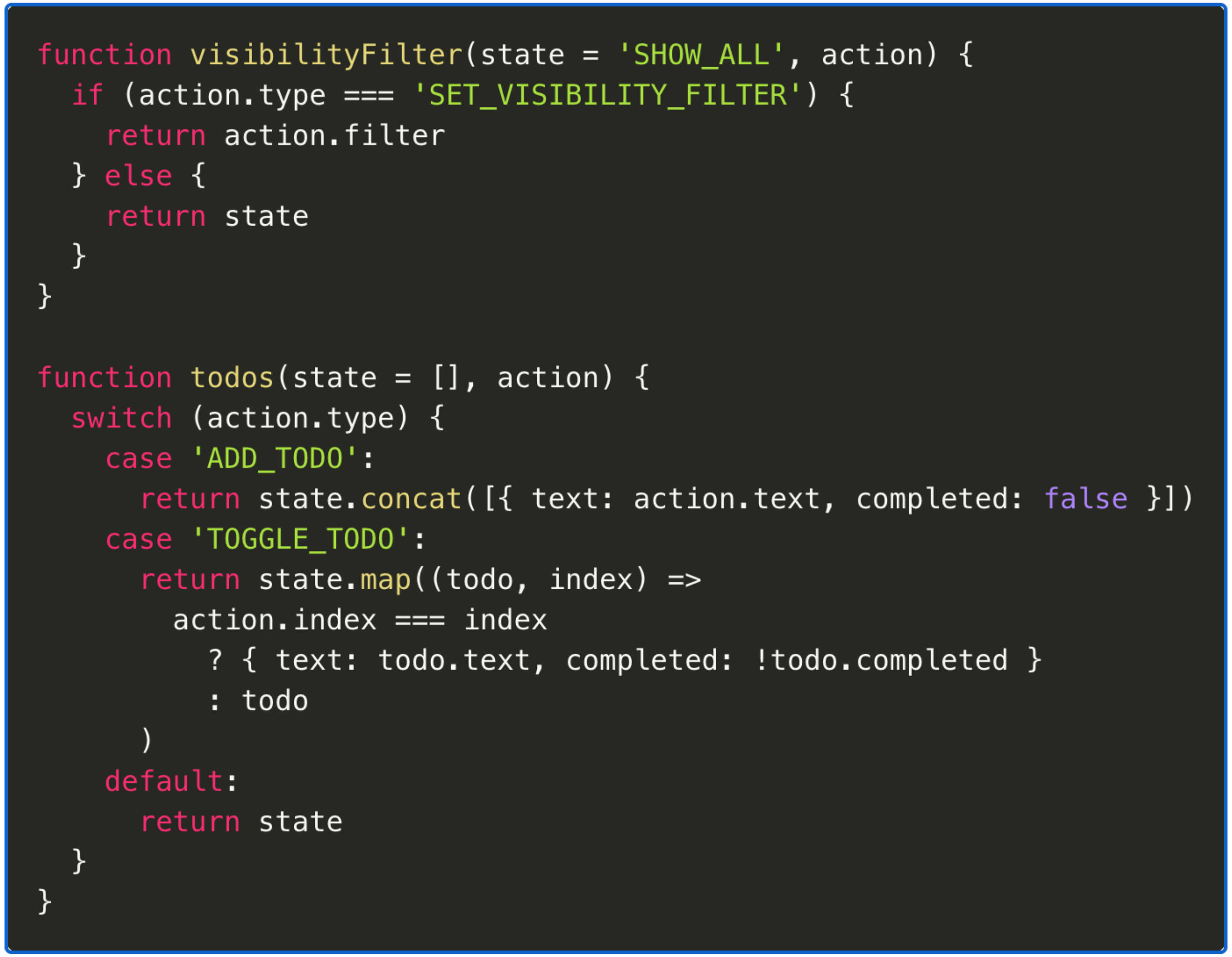
- An action is a plain JavaScript object that describes what happened



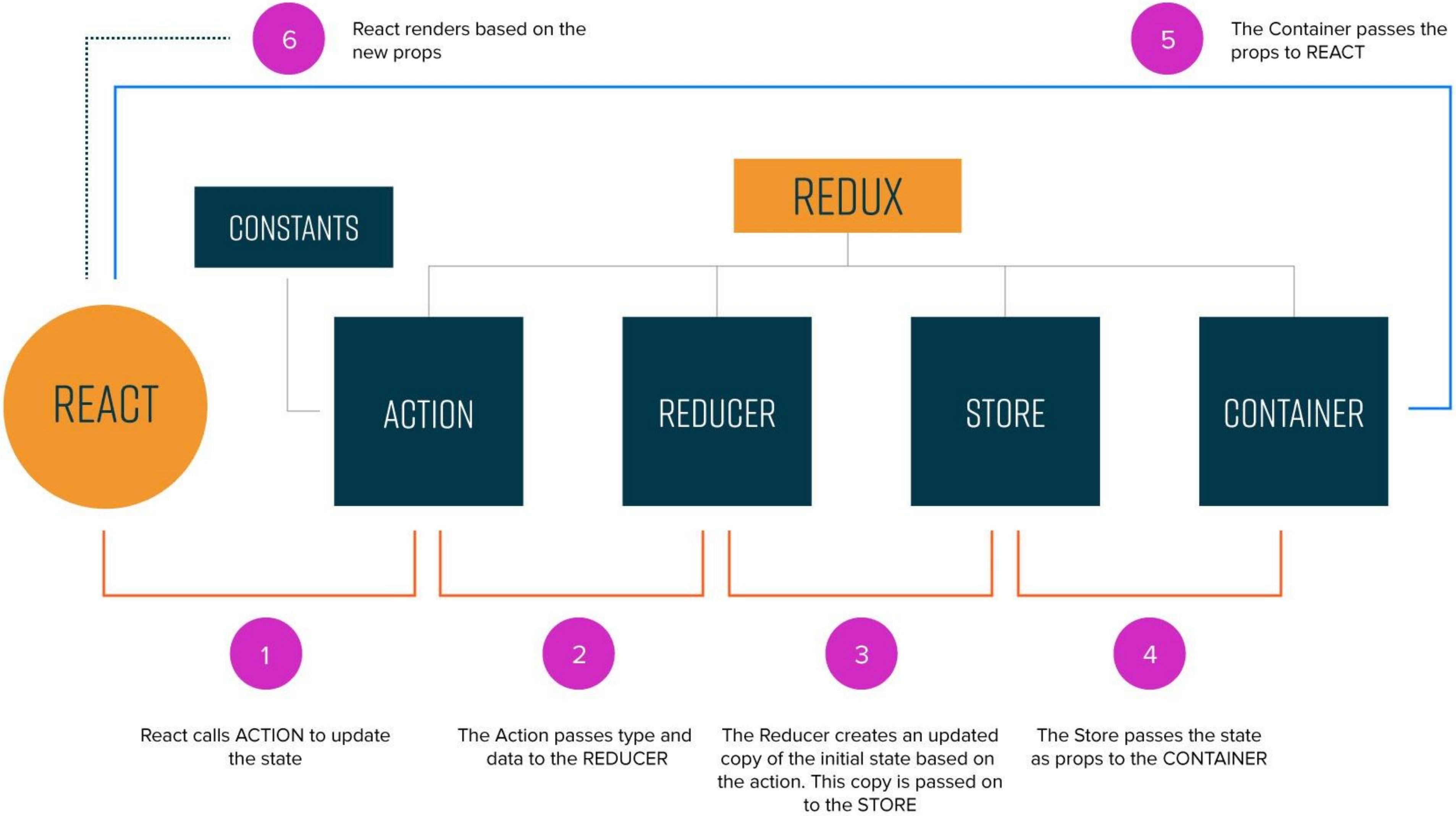
```
{ type: 'ADD_TODO', text: 'Go to swimming pool' }
{ type: 'TOGGLE_TODO', index: 1 }
{ type: 'SET_VISIBILITY_FILTER', filter: 'SHOW_ALL' }
```

React Redux - Reducer

- A function that takes state and action as arguments, and returns the next state of the app



```
function visibilityFilter(state = 'SHOW_ALL', action) {
  if (action.type === 'SET_VISIBILITY_FILTER') {
    return action.filter
  } else {
    return state
  }
}

function todos(state = [], action) {
  switch (action.type) {
    case 'ADD_TODO':
      return state.concat([{ text: action.text, completed: false }])
    case 'TOGGLE_TODO':
      return state.map((todo, index) =>
        action.index === index
          ? { text: todo.text, completed: !todo.completed }
          : todo
      )
    default:
      return state
  }
}
```



6 React renders based on the new props

5 The Container passes the props to REACT

1 React calls ACTION to update the state

2 The Action passes type and data to the REDUCER

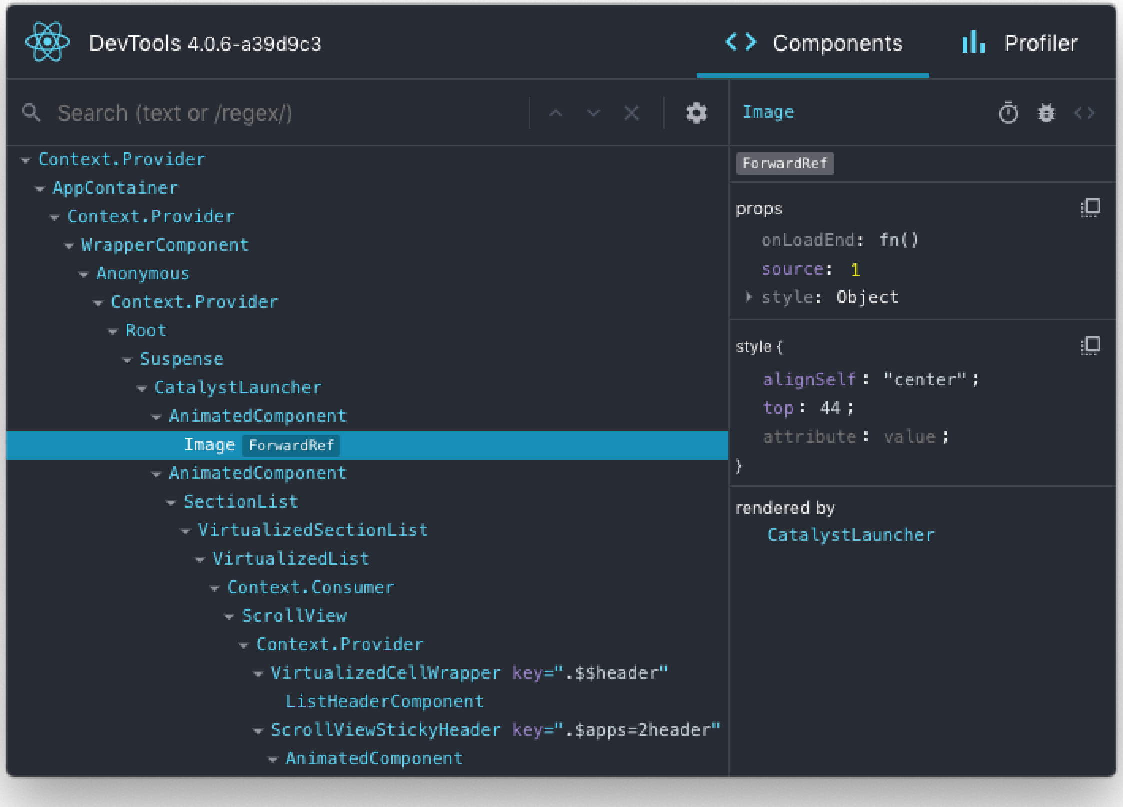
3 The Reducer creates an updated copy of the initial state based on the action. This copy is passed on to the STORE

4 The Store passes the state as props to the CONTAINER

React Devtool

- React Developer Tools is used for debugging to track props, state, tree etc. of the component.

- It allows to see the structure of the components.

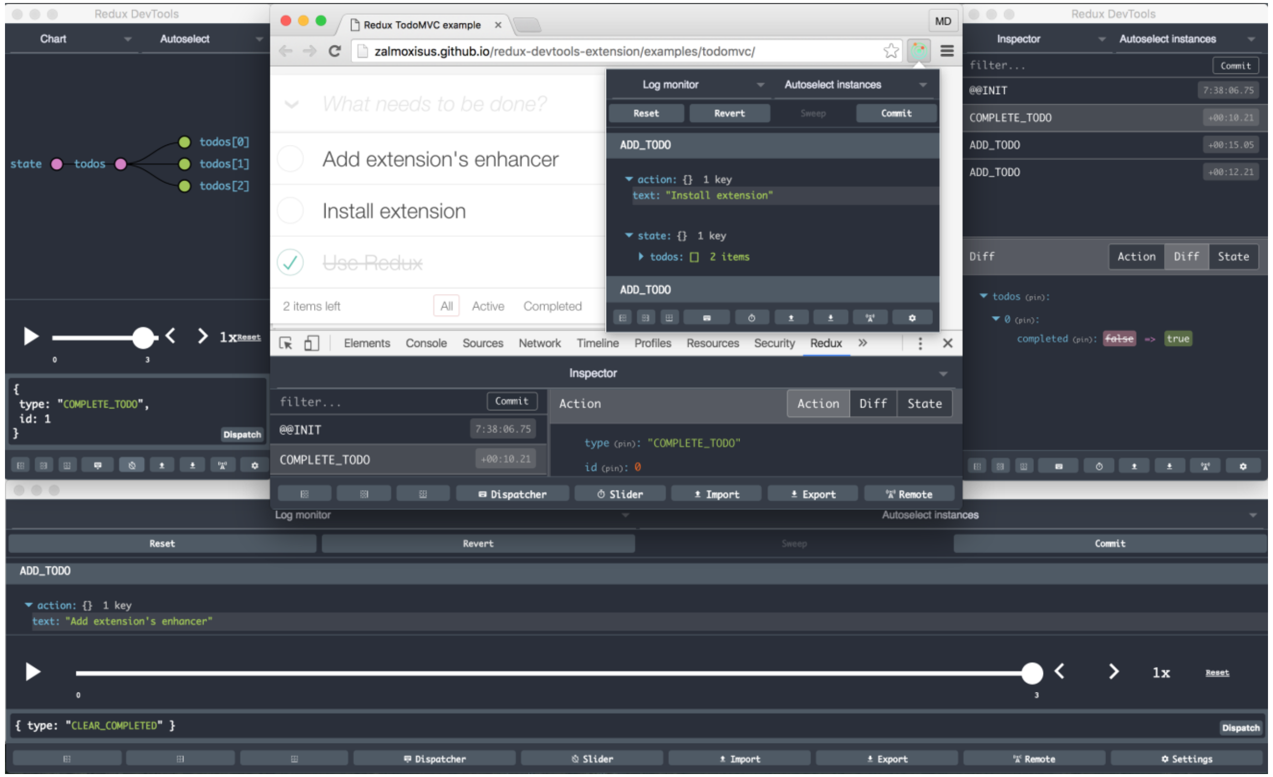


```
Context.Provider
  AppContainer
    Context.Provider
      WrapperComponent
        Anonymous
          Context.Provider
            Root
              Suspense
                CatalystLauncher
                  AnimatedComponent
                    Image ForwardRef
                  AnimatedComponent
                    SectionList
                      VirtualizedSectionList
                        VirtualizedList
                          Context.Consumer
                            ScrollView
                              Context.Provider
                                VirtualizedCellWrapper key=".$$header"
                                  ListHeaderComponent
                                ScrollViewStickyHeader key=".$apps=2header"
                                  AnimatedComponent
```

```
Image
ForwardRef

props
  onLoadEnd: fn()
  source: 1
  style: Object

style {
  alignSelf: "center";
  top: 44;
  attribute: value;
}

rendered by
  CatalystLauncher
```

Redux Devtool

- Redux DevTools is a powerful tool used to monitor redux flow, the state update process, submit dispatch action in current container.



```
{
 type: "COMPLETE_TODO",
 id: 1
}
```

```
▾ action: {} 1 key
  text: "Install extension"

▾ state: {} 1 key
  ▸ todos: [] 2 items
```

@@INIT	7:38:06.75
COMPLETE_TODO	+00:10.21
ADD_TODO	+00:15.05
ADD_TODO	+00:12.21

```
type (pin): "COMPLETE_TODO"
id (pin): 0
```

```
▾ todos (pin):
  ▾ 0 (pin):
    completed (pin): ~~false~~ => true
```

```
▾ action: {} 1 key
  text: "Add extension's enhancer"
```

```
{ type: "CLEAR_COMPLETED" }
```

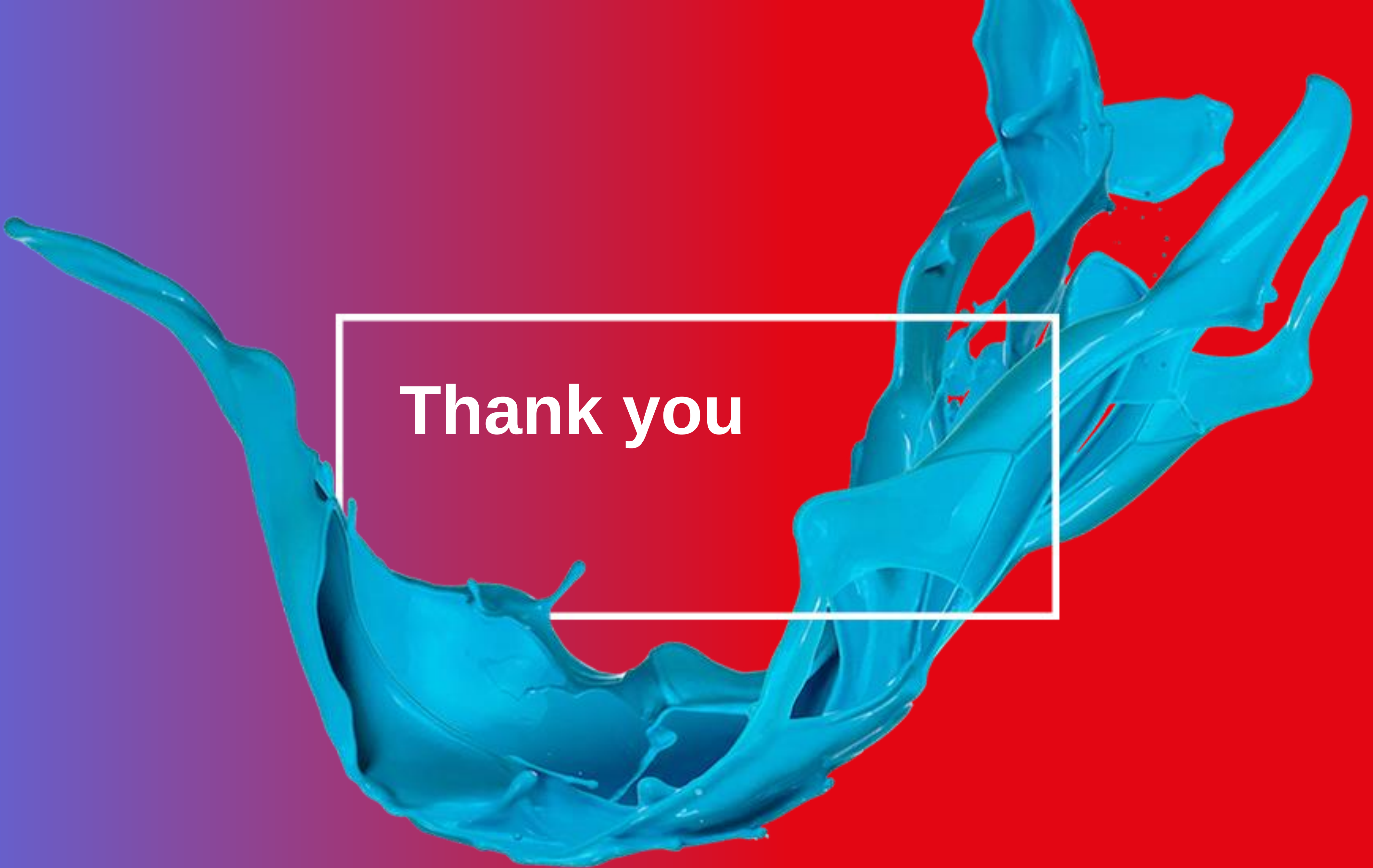
Redux Saga & Redux Thunk

- Redux Thunk:
  - Redux-Thunk middleware allows you to write action creators that return a function instead of the typical action object.
  - The thunk can then be used to delay the dispatch of an action until the fulfillment of an asynchronous line of code (e.g., an axios request to receive data).

- Redux Saga:
  - Redux-Saga is a library that aims to make application side effects (e.g., asynchronous actions such as fetching data) easier to handle and more efficient to execute.
  - Like Redux-Thunk, Redux-Saga has access to the full Redux application state and it can dispatch Redux actions as well.

Exercise

- Implement a new page & apply Redux
- Use Redux Thunk in the application



Thank you