React + Redux

case study

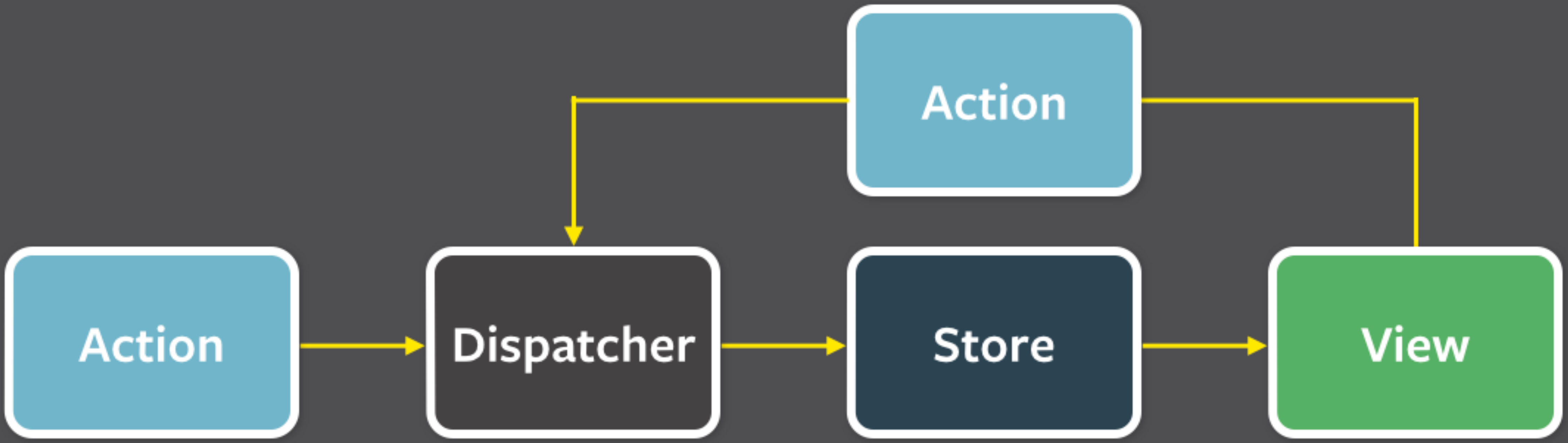
Adam Babik
@dreame4

sauce LABS

Review

- React — JS library for building user interfaces
- Flux — the application architecture
- Redux — a predictable state container

Flux



Redux

- Reduces complexity in comparison with Flux
- Simple API
- Tiny (2kB) and independent from React
- Has extremely good docs and video tutorials

Redux three principles

ONE STORE
TO RULL THEM ALL
imgflip.com

State is **read-only** and is changed by dispatching an action

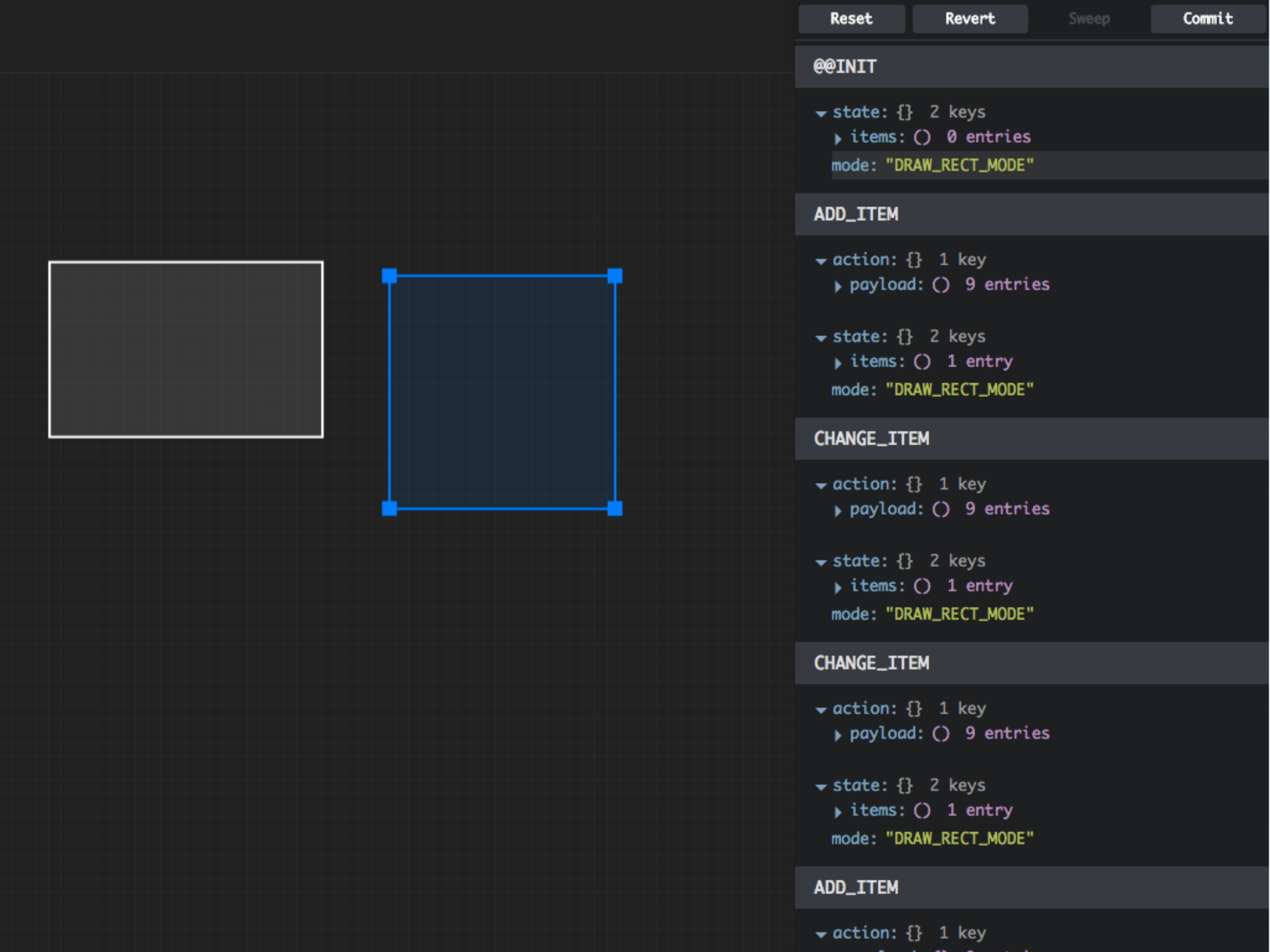
State transformations are described using **pure functions**

Reducer Example

```
function todoApp(state = initialState, action) {
  switch (action.type) {
    case SET_VISIBILITY_FILTER:
      return Object.assign(
        {},
        state,
        { visibilityFilter: action.filter }
      );
    default:
      return state;
  }
}
```

Demo

github.com/dreame4/drawapp-react-redux



```
@@INIT

▾ state: {}  2 keys
  ▸ items: ()  0 entries
  mode: "DRAW_RECT_MODE"

ADD_ITEM

▾ action: {}  1 key
  ▸ payload: ()  9 entries

▾ state: {}  2 keys
  ▸ items: ()  1 entry
  mode: "DRAW_RECT_MODE"

CHANGE_ITEM

▾ action: {}  1 key
  ▸ payload: ()  9 entries

▾ state: {}  2 keys
  ▸ items: ()  1 entry
  mode: "DRAW_RECT_MODE"

CHANGE_ITEM

▾ action: {}  1 key
  ▸ payload: ()  9 entries

▾ state: {}  2 keys
  ▸ items: ()  1 entry
  mode: "DRAW_RECT_MODE"

ADD_ITEM

▾ action: {}  1 key
```

Redux Examples

Create Store

```
const store = createStore(
  rootReducer,
  initialState
);

// create a store with middleware
const createStoreWithMiddleware = compose(
  applyMiddleware(...middleware),
  DevTools.instrument()
)(createStore);
const store = createStoreWithMiddleware(
  rootReducer,
  initialState
);
```

Create & Dispatch An Action

```
function addItem(item) {
  return {
    type: "ADD_ITEM",
    payload: item
  };
}

// send the action
store.dispatch(addItem(item));
```

Update State

```
let initialState = [];

function items(state = initialState, action) {
  switch (action.type) {
    case "ADD_ITEM":
      return state.concat([action.payload]);
    default:
      return state;
  }
}
```

Connect With React

```
const createSmartComponent = connect(
  // map state to props
  state => ({
    mode: state.mode,
    items: state.items
  }),
  // map actions to props
  dispatch => bindActionCreators({
    changeMode,
    saveState
  }, dispatch)
);

export default createSmartComponent(Toolbar);
```

Links

- Redux docs and tutorial: **redux.js.org**
- Redux video tutorial: **egghead.io/series/getting-started-with-redux**
- Starter kit or boilerplate:
  - github.com/davezuko/react-redux-starter-kit
  - github.com/tj/frontend-boilerplate

A CARTOON INTRO TO REDUX

Thanks!

Questions?