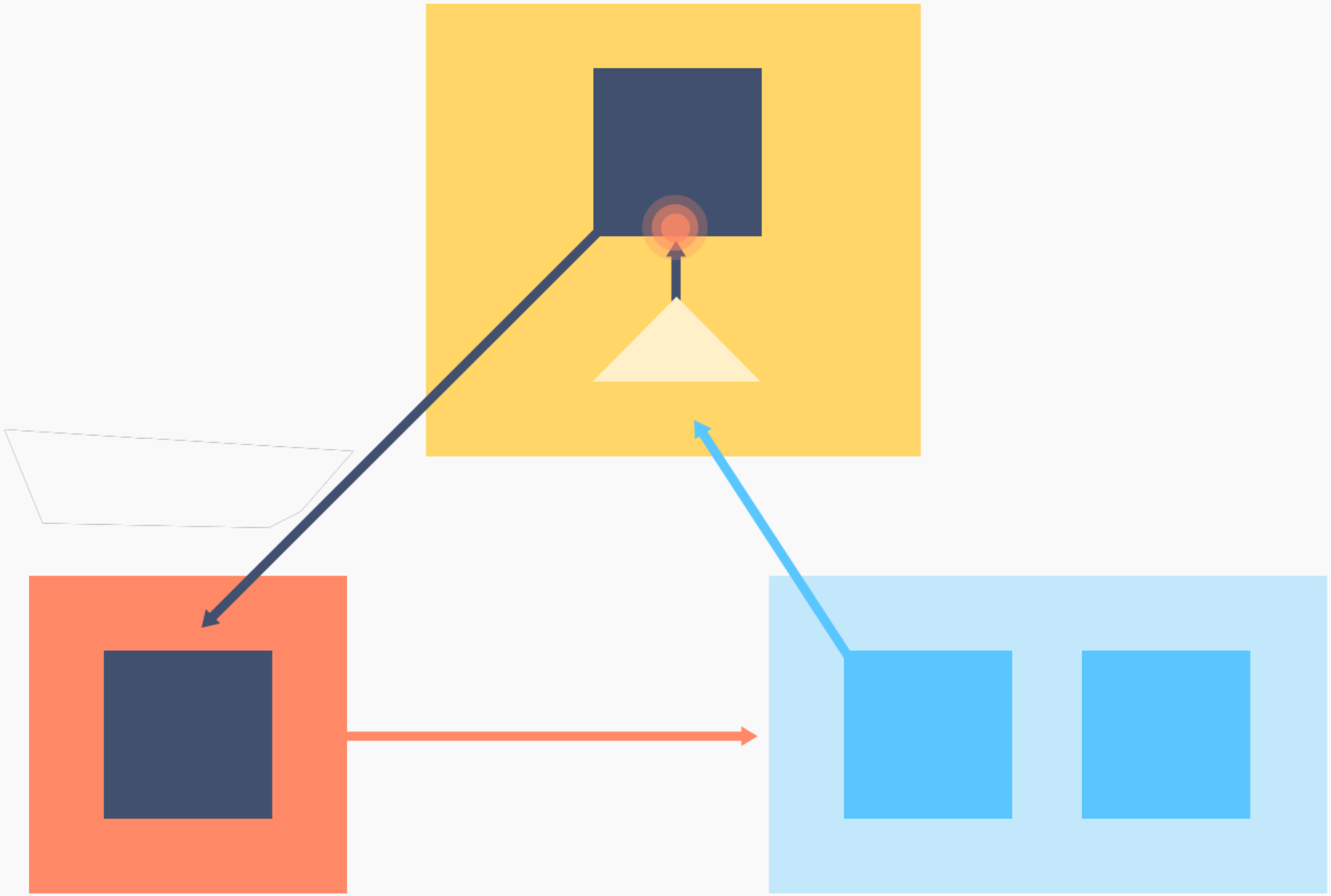
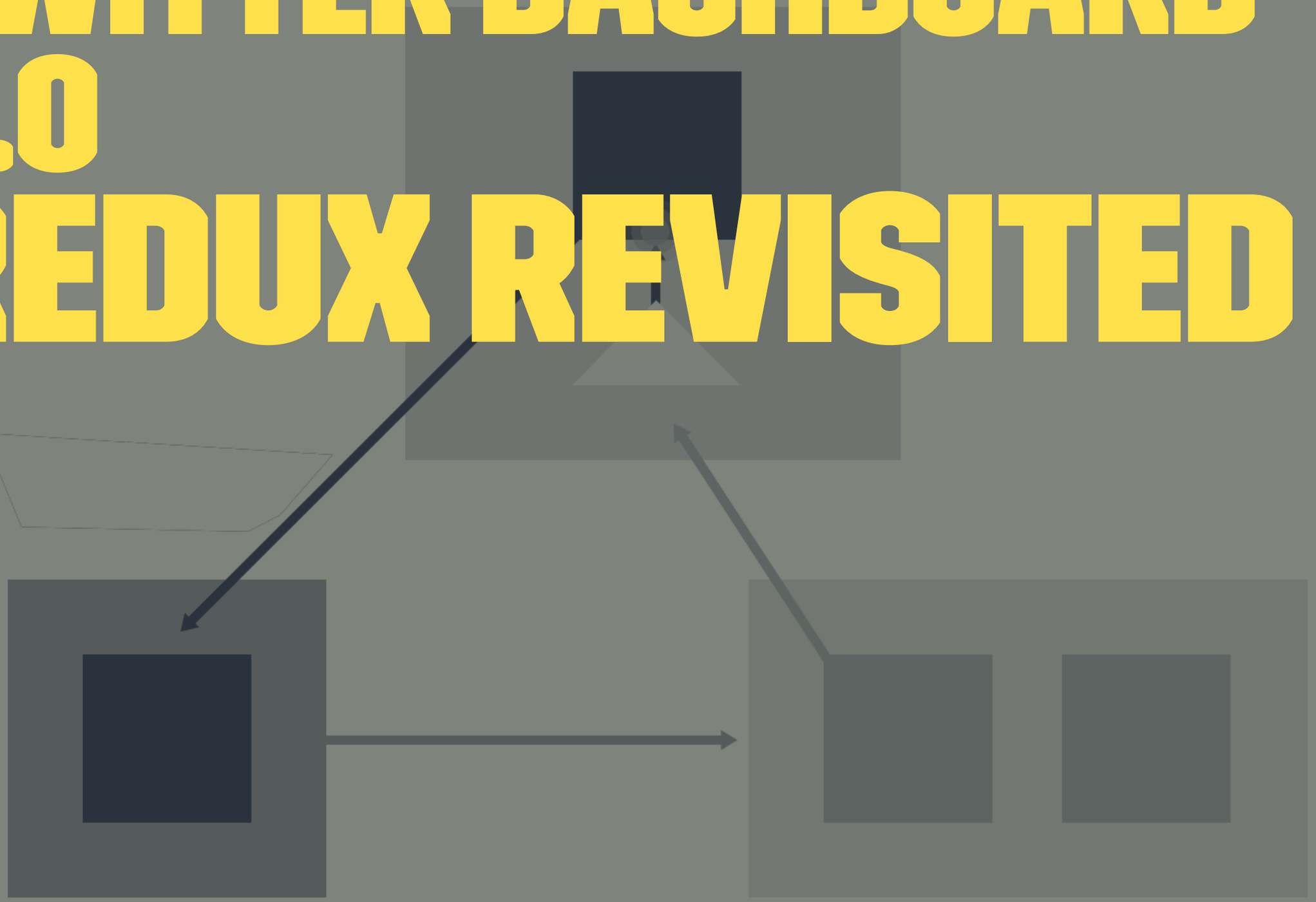
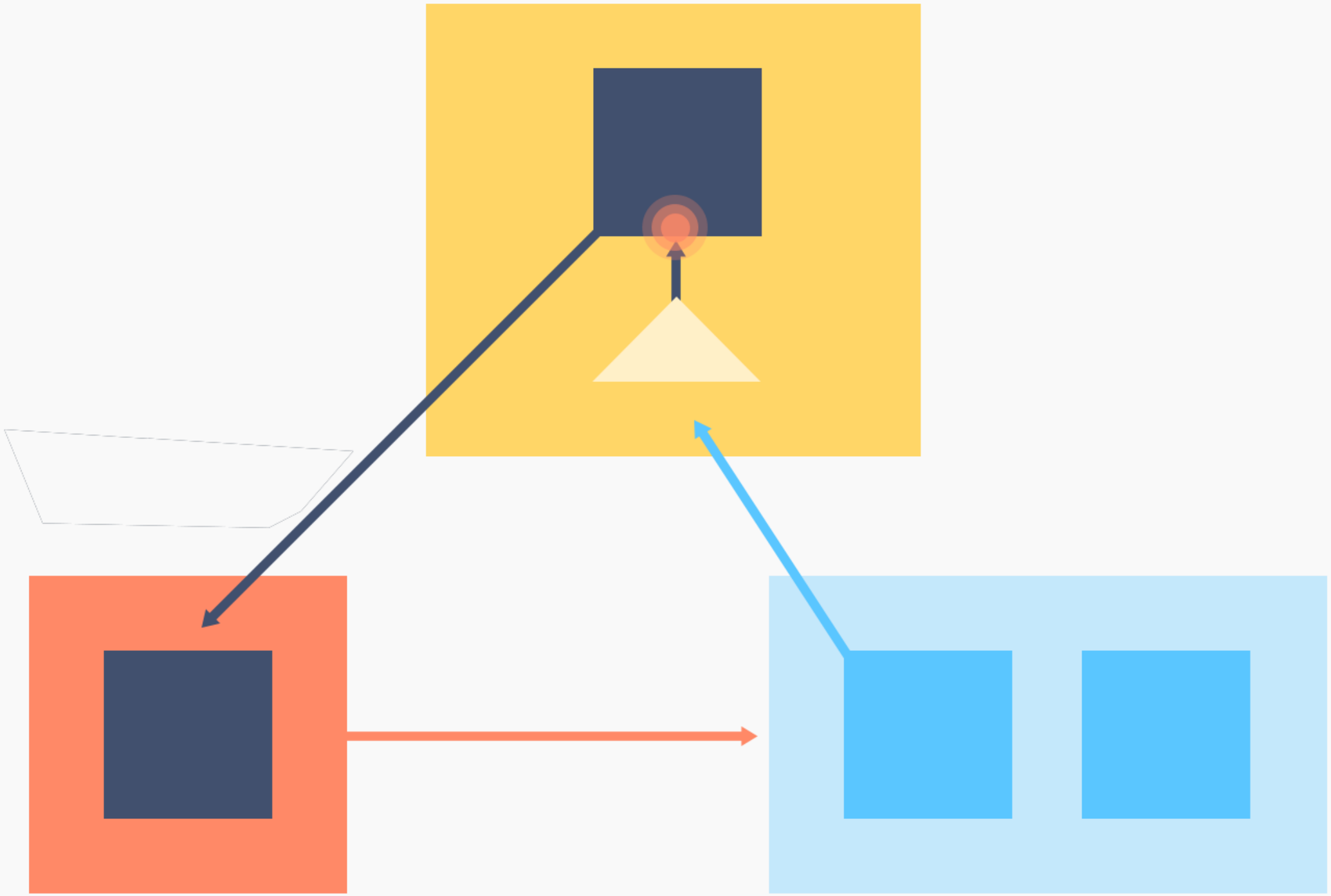


TWITTER DASHBOARD 2.0 REDUX REVISITED

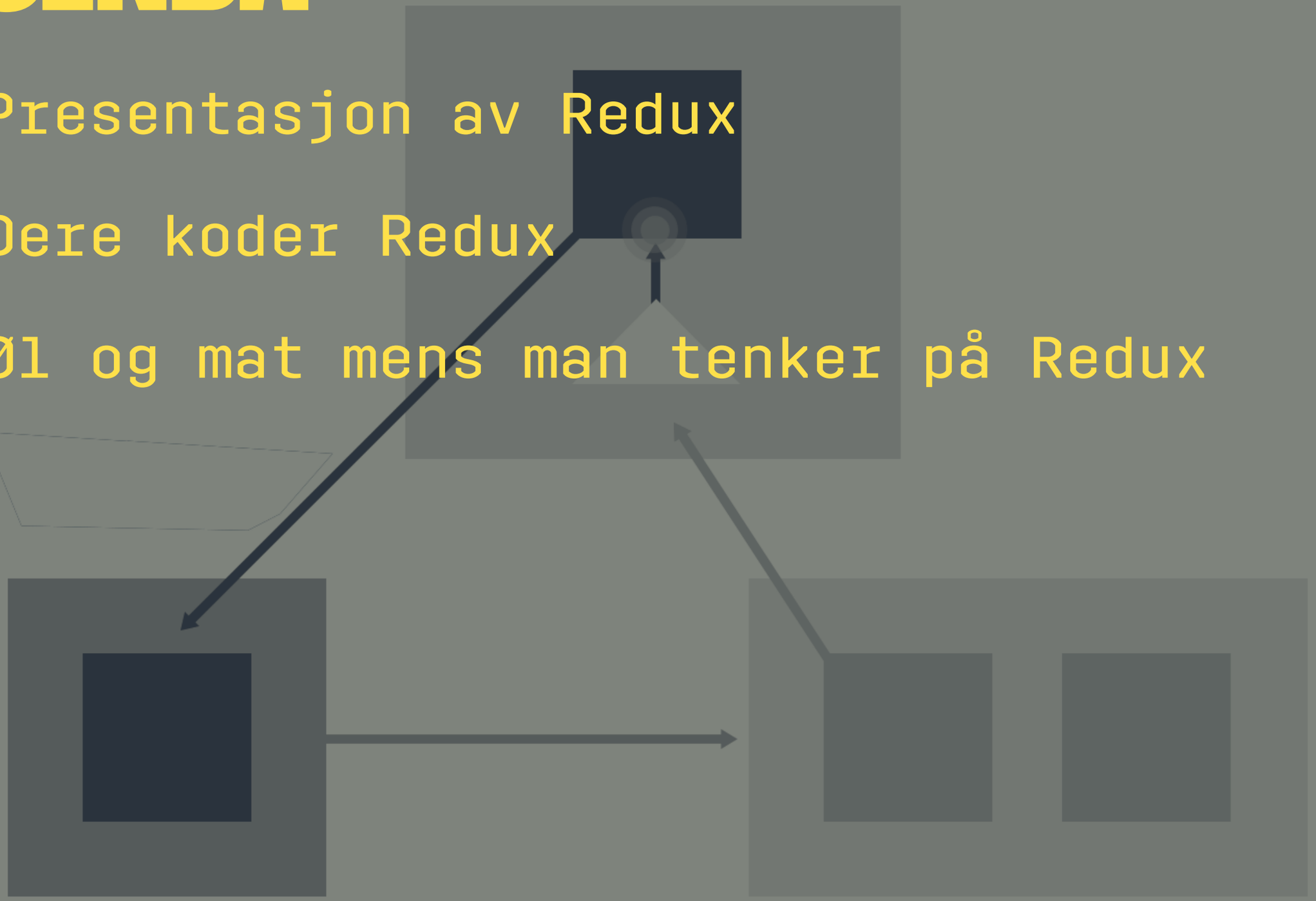


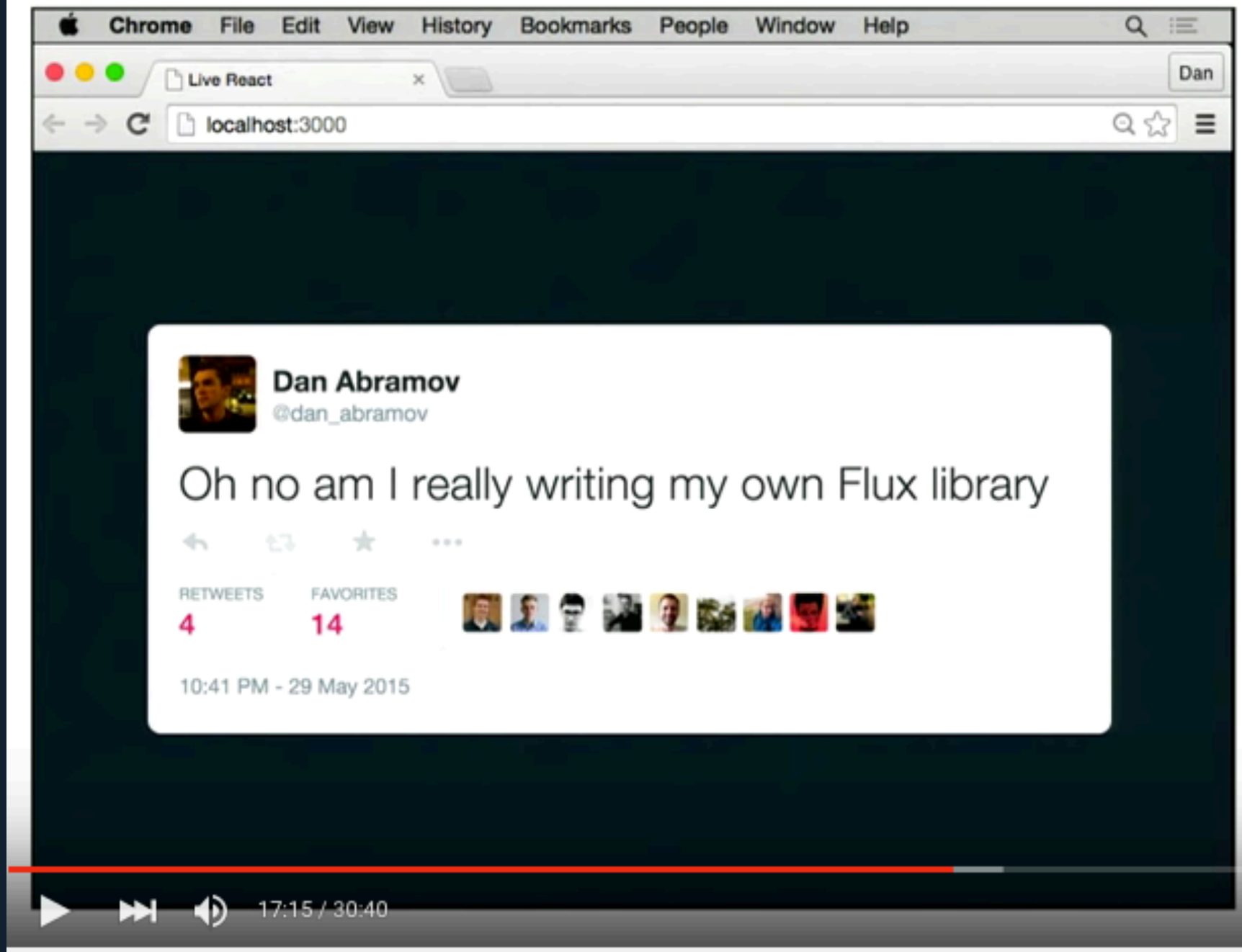
// GURO SETERNES, INGAR ALMKLOV, ERIK WENDEL

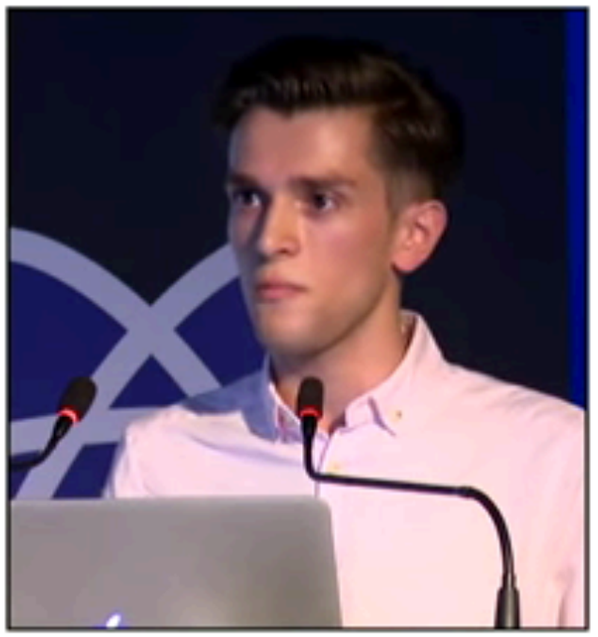


AGENDA

1. Presentasjon av Redux
2. Dere koder Redux
3. Øl og mat mens man tenker på Redux







Dan Abramov - Live React: Hot Reloading with Time Travel at react-europe 2015

ReactEurope

Subscribe 4,779

130,362

Add to Share More

2,087 4

todo

What needs to be done?

Use Redux

1 item left

All Active Com

Reset Revert Sweep Commit

@@INIT

state: {} 1 key

todos: [] 1 item

reactjs / redux

Watch 796 Star 16,997 Fork 2,069

Predictable state container for JavaScript apps http://redux.js.org

1,914 commits **8** branches **45** releases **283** contributors

Branch: **master** New pull request New file Upload files Find file SSH git@github.com:reactjs/redux Download ZIP

gaearon Update Ecosystem.md Latest commit 1e517a3 a day ago

build	Use lodash-es in ES Modules build	2 months ago
docs	Update Ecosystem.md	a day ago
examples	fix minor typo	a month ago
src	Avoid every execution and allocation (#1595)	3 days ago
test	Fix typescript test of store enhancer	10 days ago
.babelrc	Use lodash-es in ES Modules build	2 months ago
.editorconfig	editorconfig: do not trim trailing whitespaces in Markdown files	2 months ago
.eslintignore	Really ignore all node_modules and dist in eslint.	a month ago
.eslintrc	Bump eslint version	6 months ago
.flowconfig	Add Flow type annotations	9 months ago
.gitignore	Add ES Modules build	2 months ago
.travis.yml	Cache examples node_modules.	a month ago
CHANGELOG.md	rackt -> reactjs (see #1392)	2 months ago
CNAME	Ensure CNAME survives doc rebuilds	6 months ago
CODE_OF_CONDUCT.md	Bump Contributor Covenant to 1.4	2 months ago

KVALITETER

» funksjonelt inspirert

» svært testbart

» klar fordeling av ansvar

» egnet for store kodebaser med lang levetid

THE GIST

» The whole state of your app is stored in an object tree inside a single store.

» The only way to change the state tree is to emit an action, an object describing what happened.

» To specify how the actions transform the state tree, you write pure reducers.

redux.js.org

THE GIST

» The whole state of your app is stored in an object tree inside a single store.

» The only way to change the state tree is to emit an action, an object describing what happened.

» To specify how the actions transform the state tree, you write pure reducers.

redux.js.org

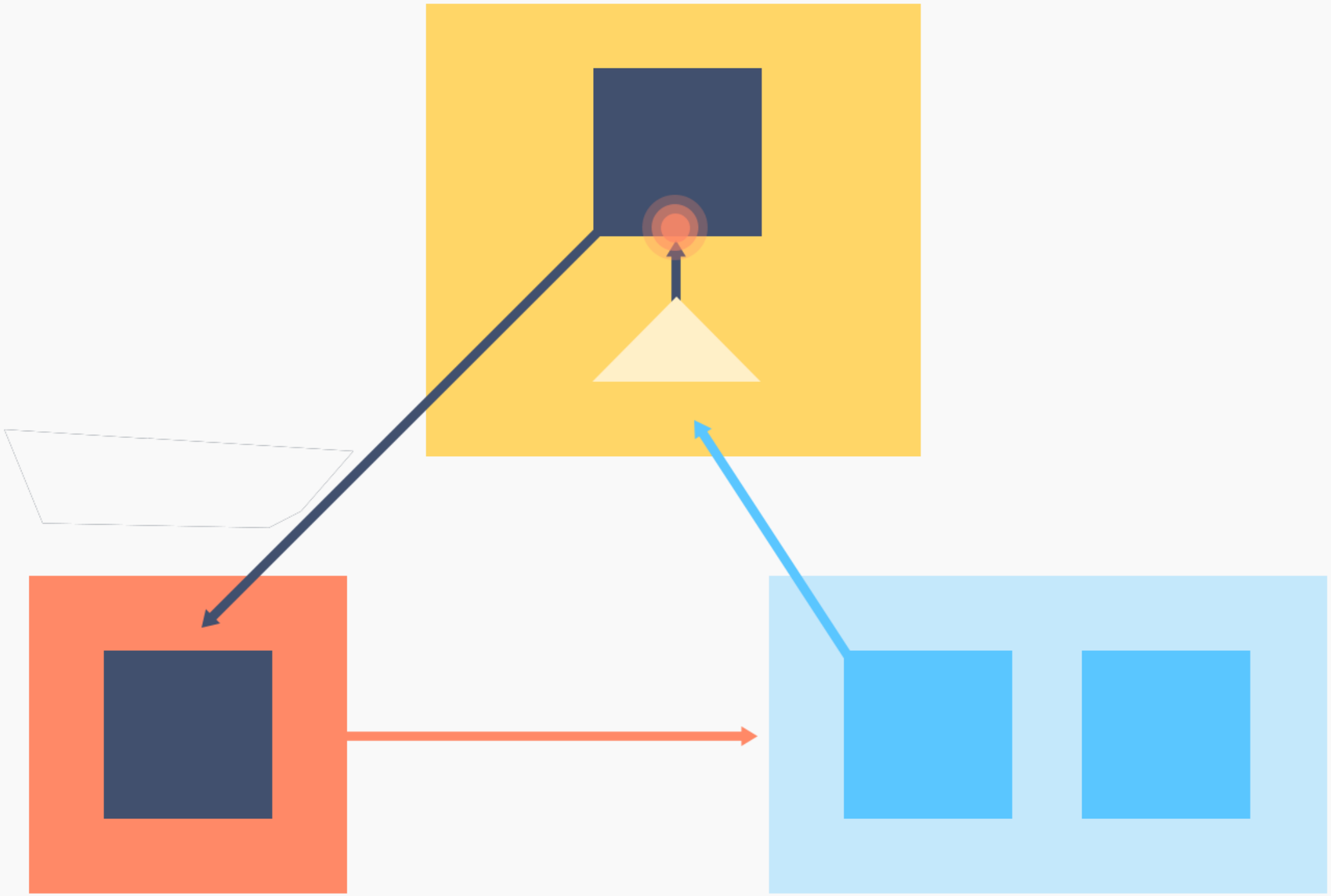
THE GIST

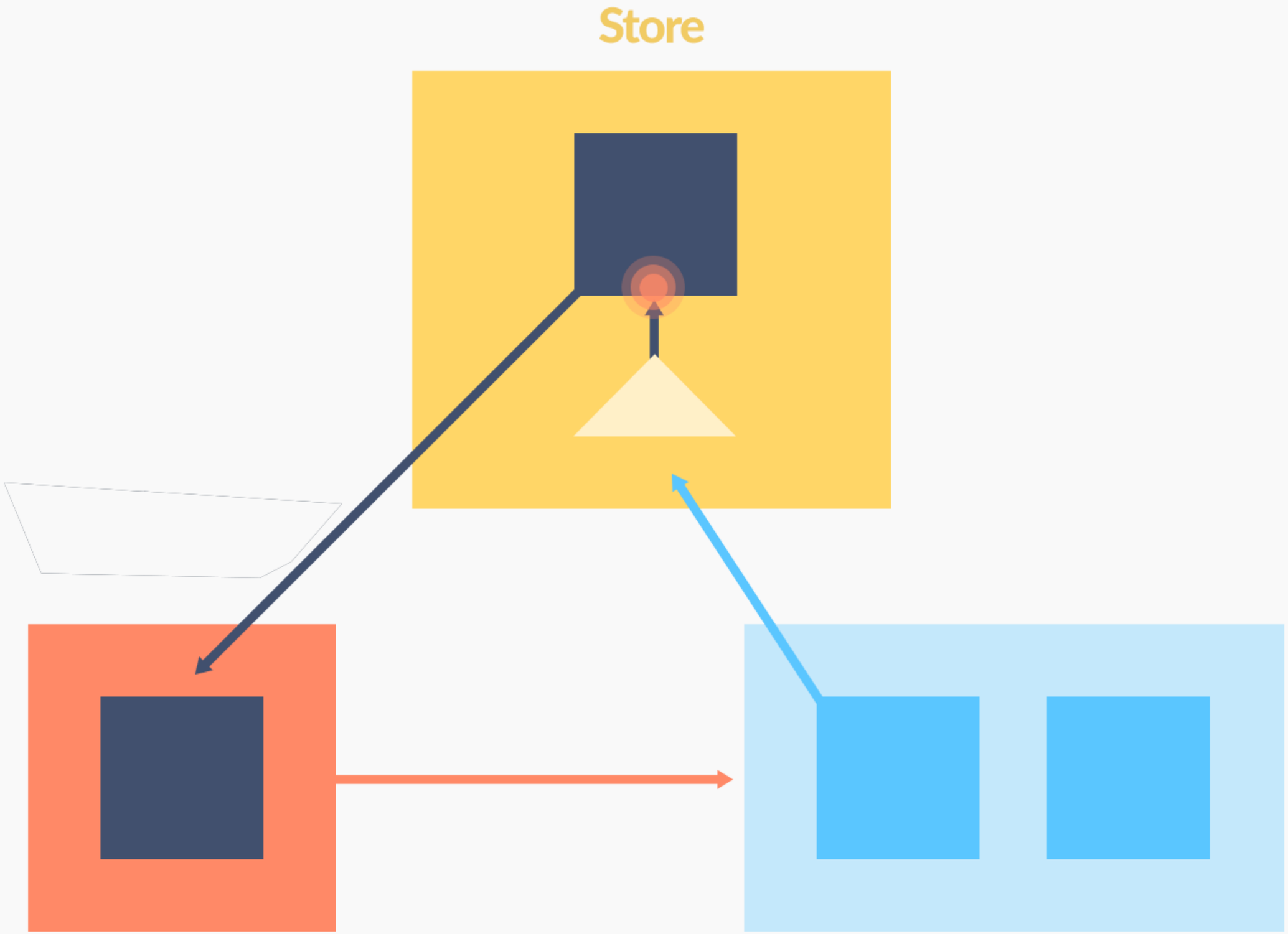
» The whole state of your app is stored in an object tree inside a single store.

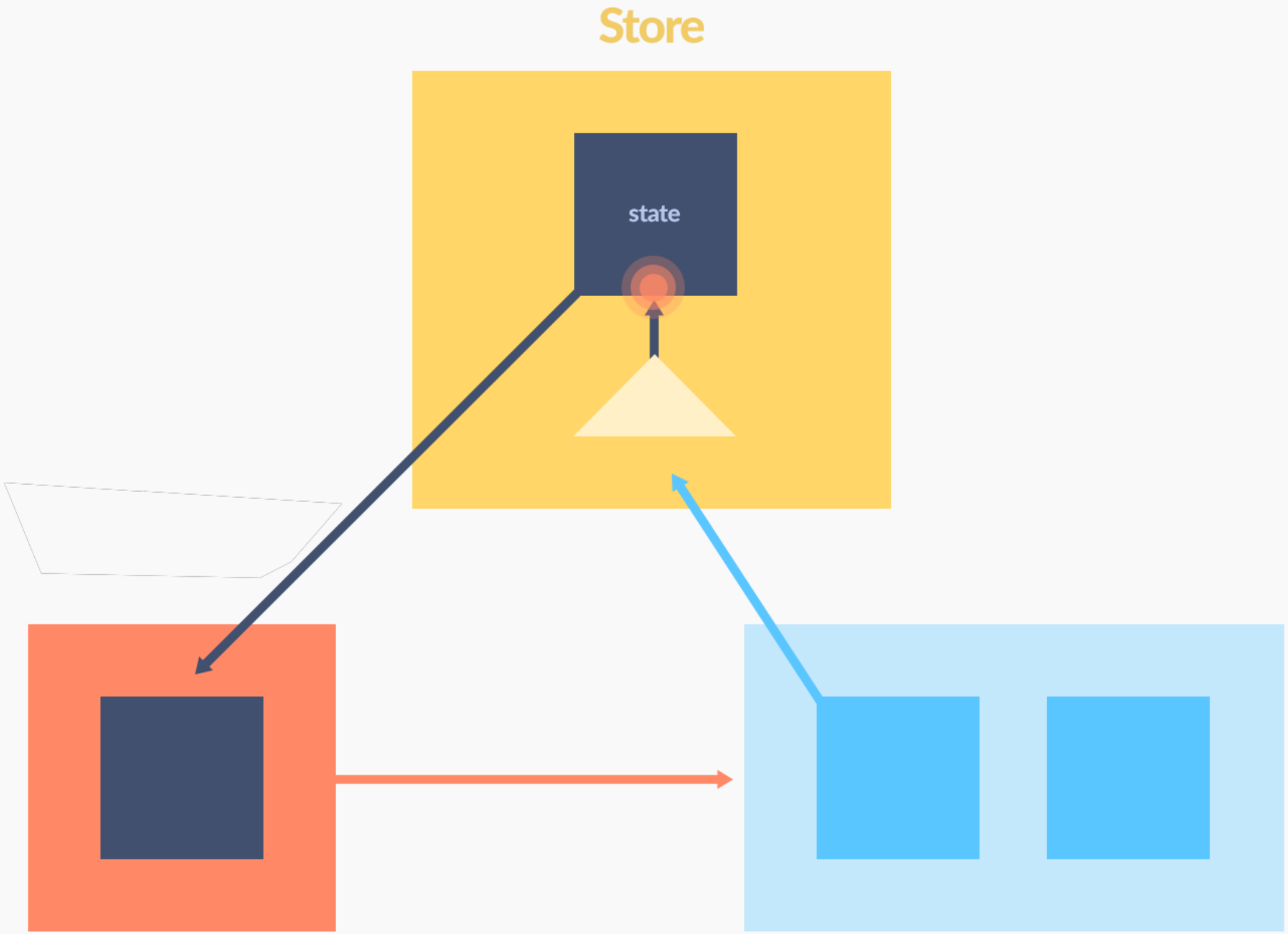
» The only way to change the state tree is to emit an action, an object describing what happened.

» To specify how the actions transform the state tree, you write pure reducers.

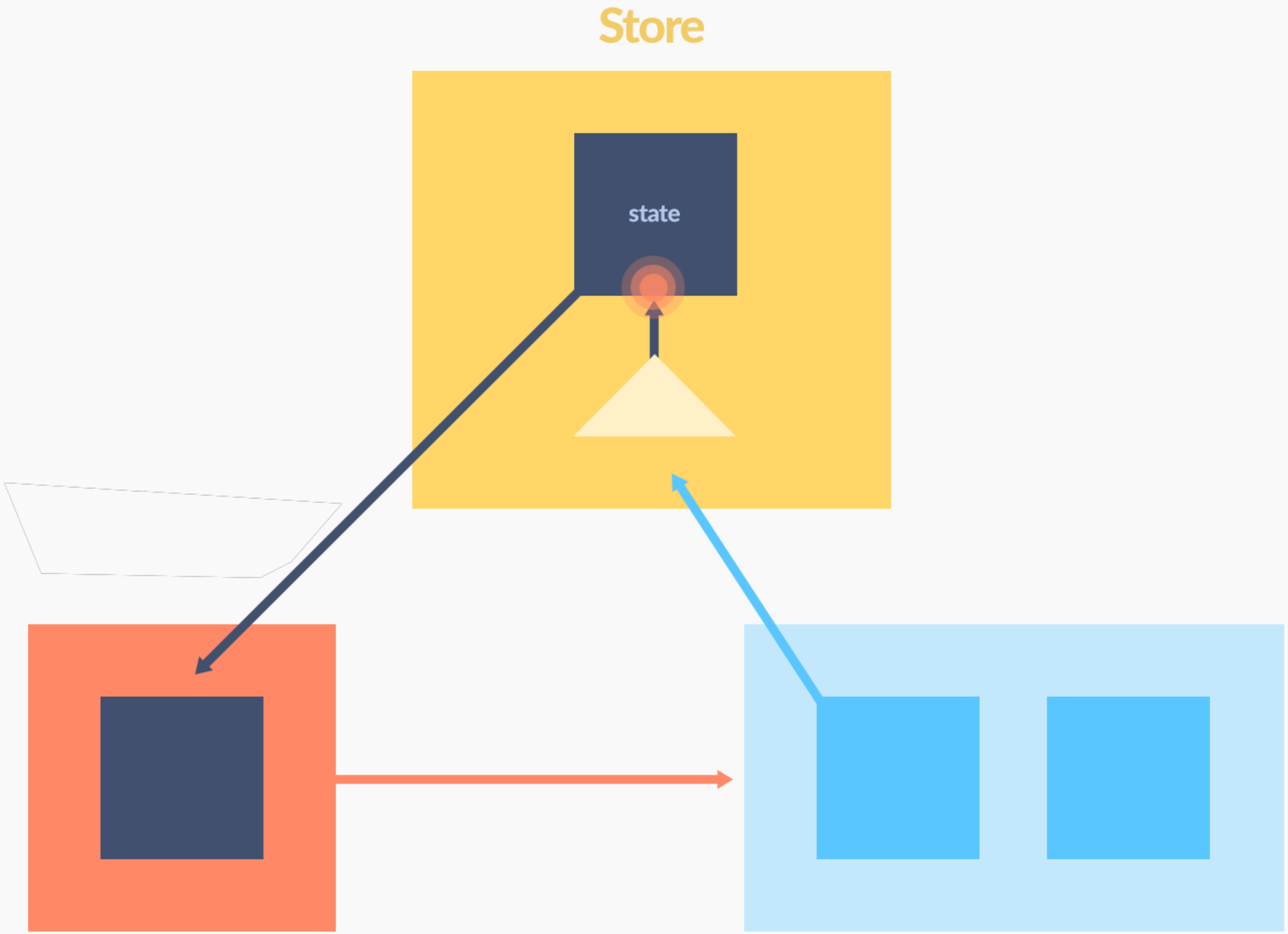
redux.js.org

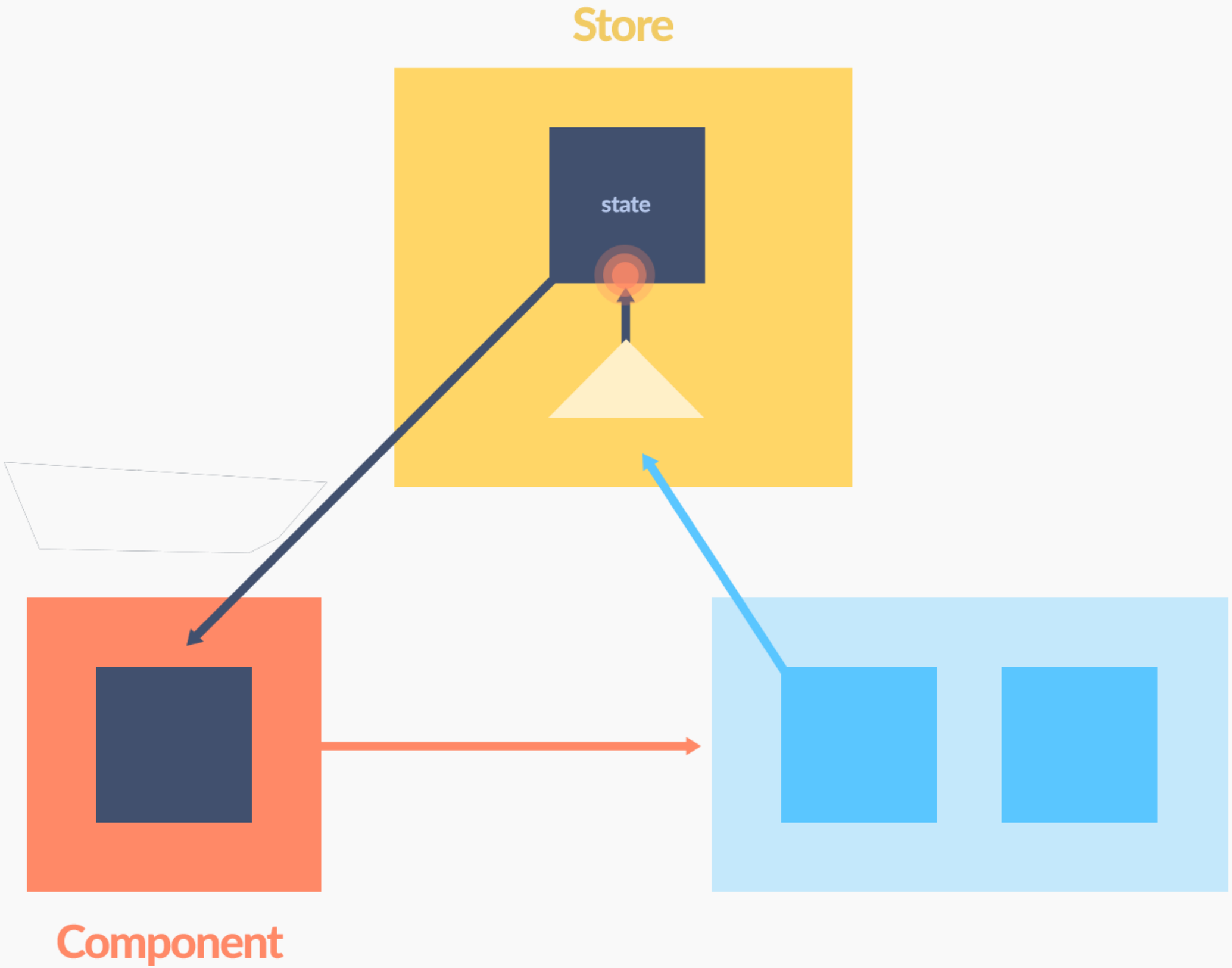


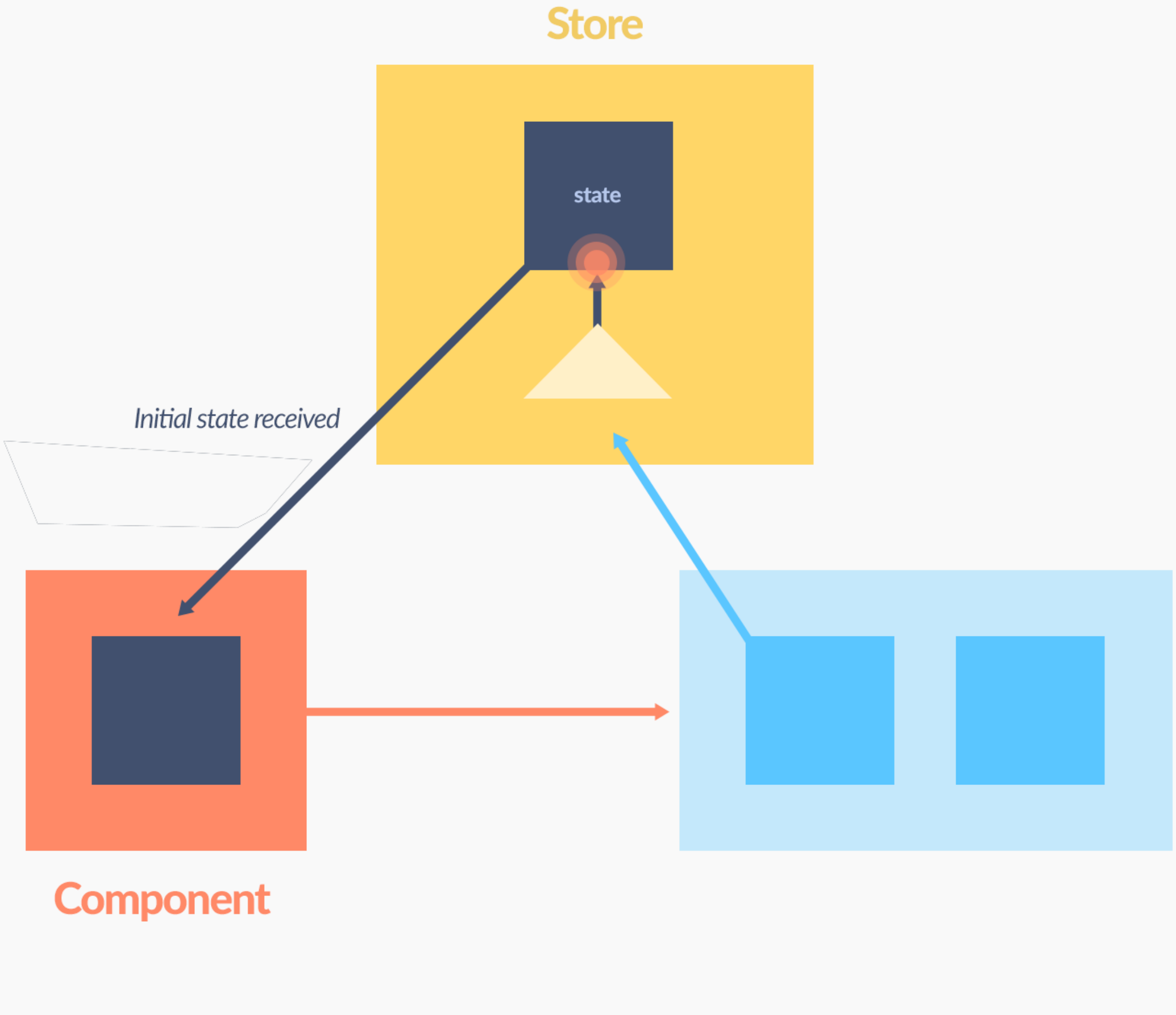

Store

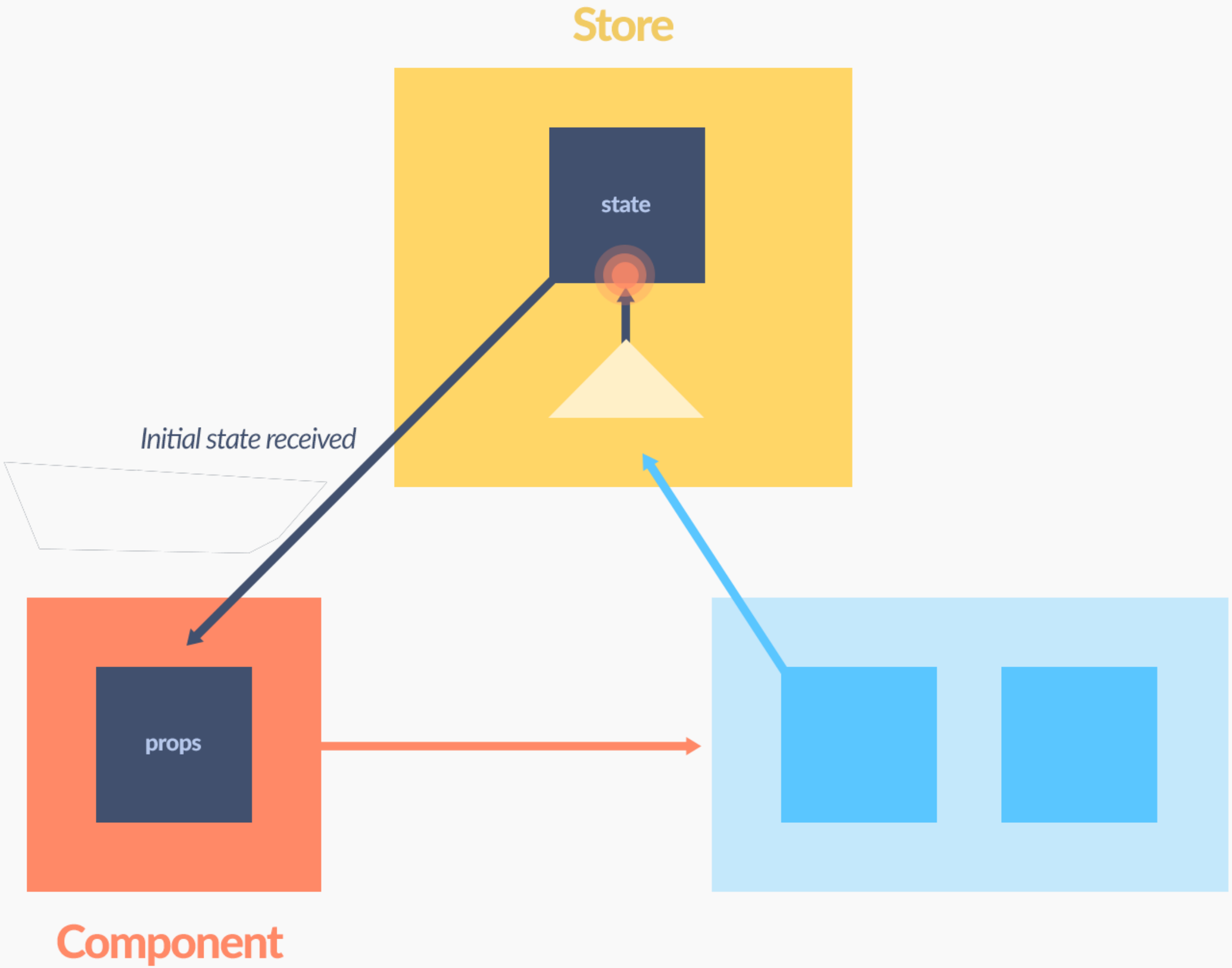

Store
state

```
{
    todos: [],
    activeFilter: 'completed',
}
```


Store
state


Store
state
Component


Store
state
Initial state received
Component


Store
state
Initial state received
props
Component

React

RE-RENDER
ALL THE THINGS
memegenerator.net

DATA => HTML

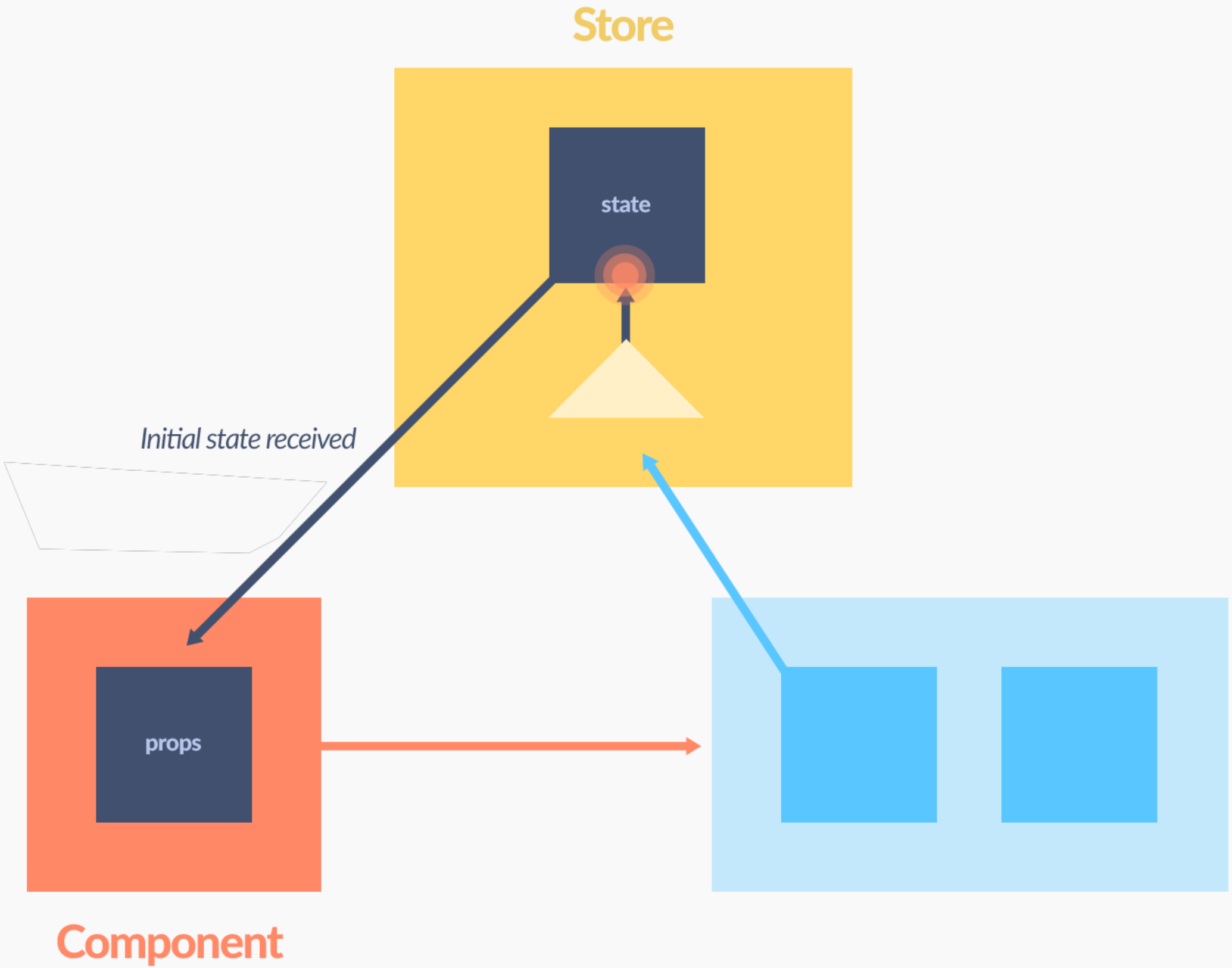
```
function SomeComponent() {
    return <div>
        Hello World!
    </div>
}
```

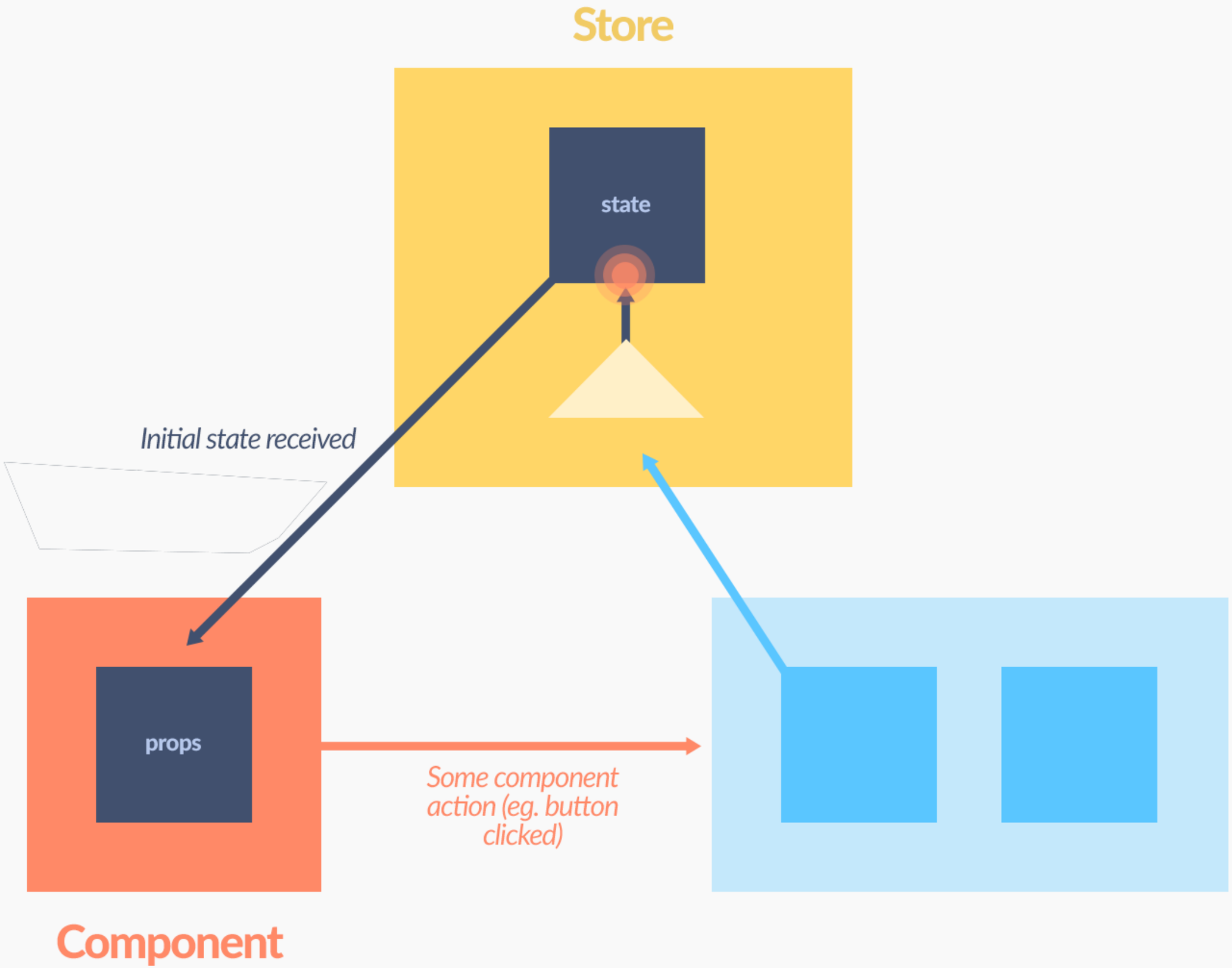
```
import { render } from 'react-dom';

render(<SomeComponent />, document.body);
```

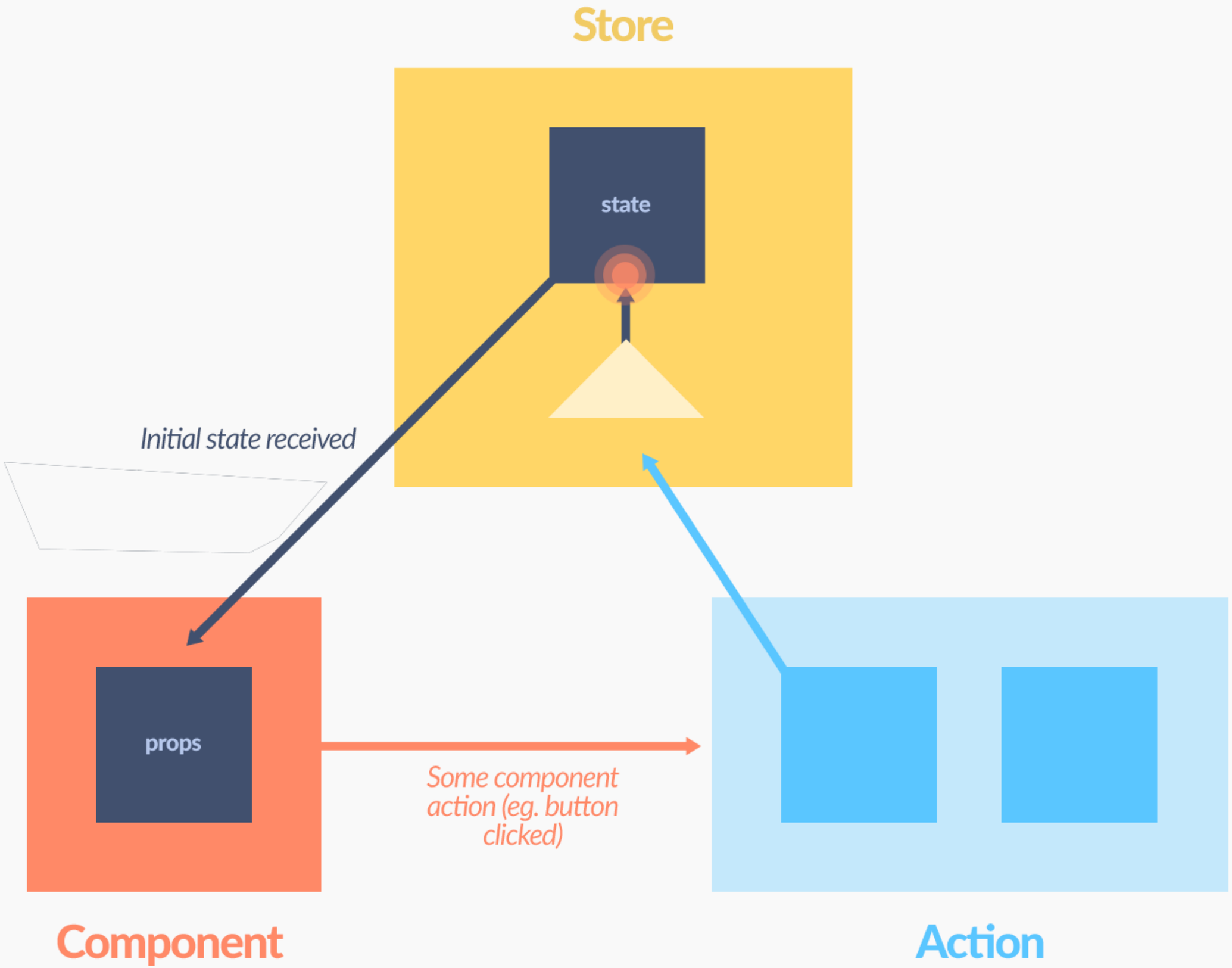
```
function SomeComponent(props) {
    return <div>
        Hello {props.name}!
    </div>
}

render(<SomeComponent name="World!" />, document.body);
```

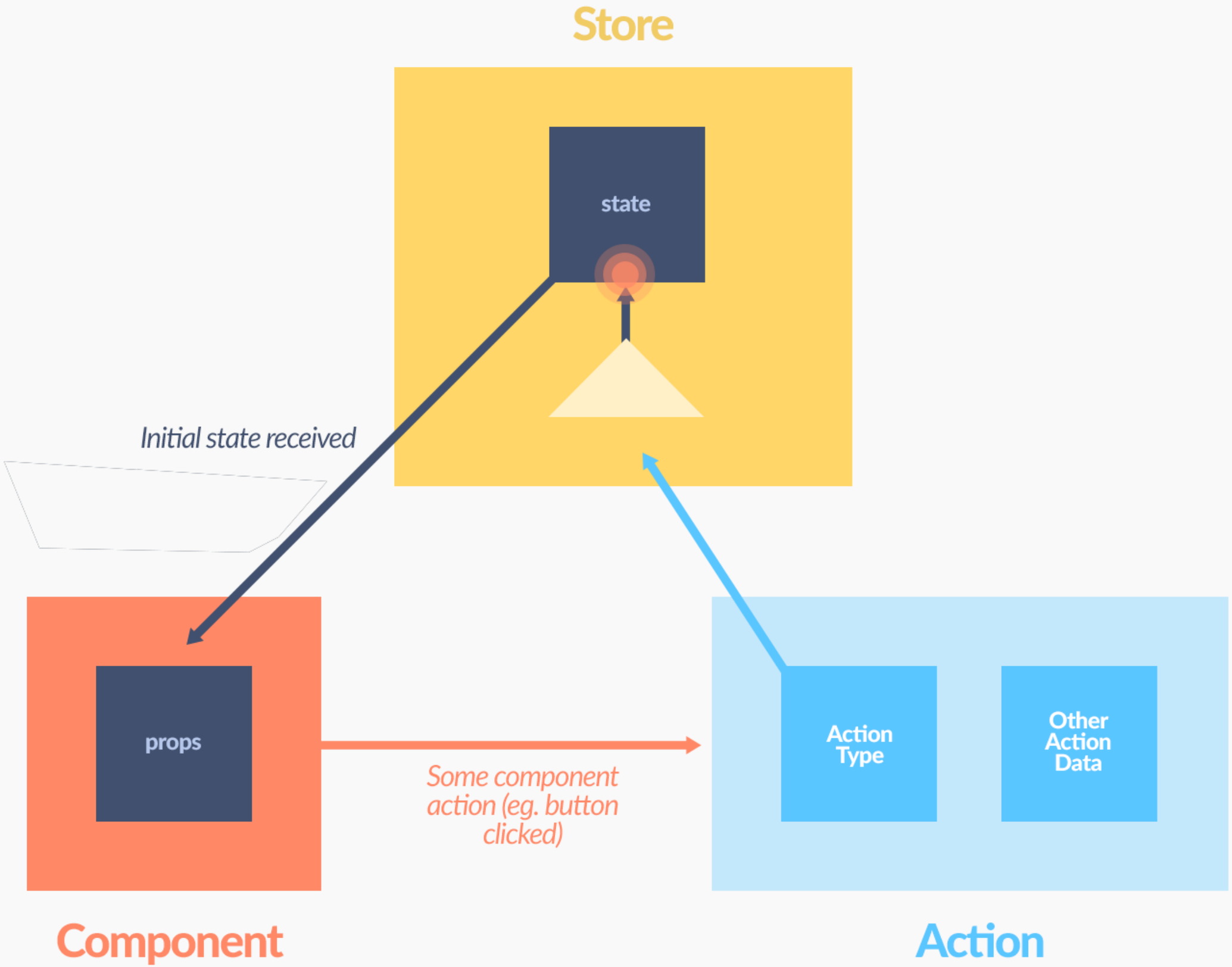
```
function SomeComponent() {
    return <div>
            <OtherComponent />
            <YetAnotherComponent />
        </div>;
}
```

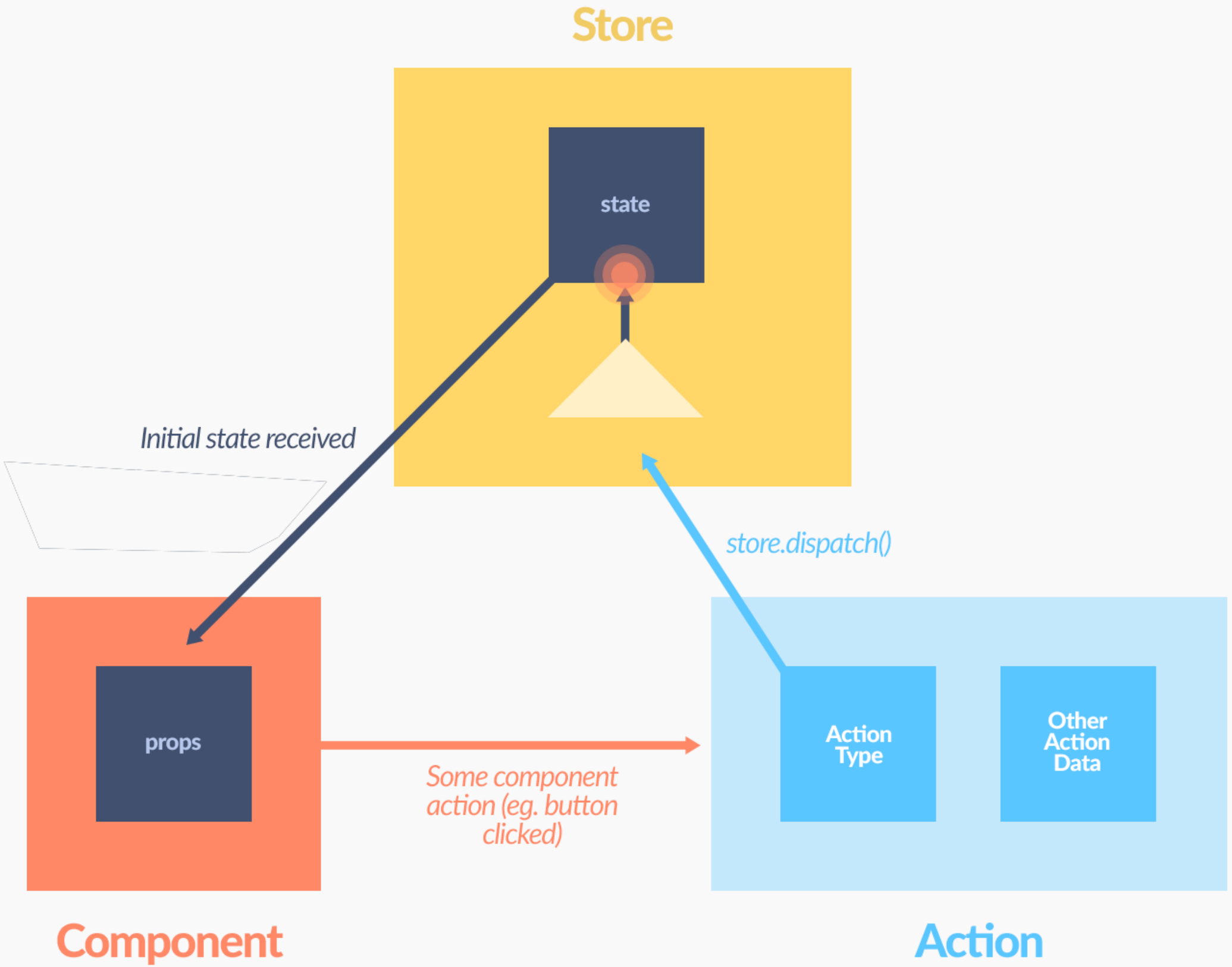
```
test(function() {
    const wrapper = wrap(<SomeComponent foo="bar" />);
    const itemCount = wrapper.find('.item').length;
    assert.equal(itemCount, 3)
})
```


Store
state
Initial state received
props
Component


Store
state
Initial state received
props
Some component action (eg. button clicked)
Component


Store
state
Initial state received
props
Some component action (eg. button clicked)
Component
Action


Store
state
Initial state received
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action


Store
state
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

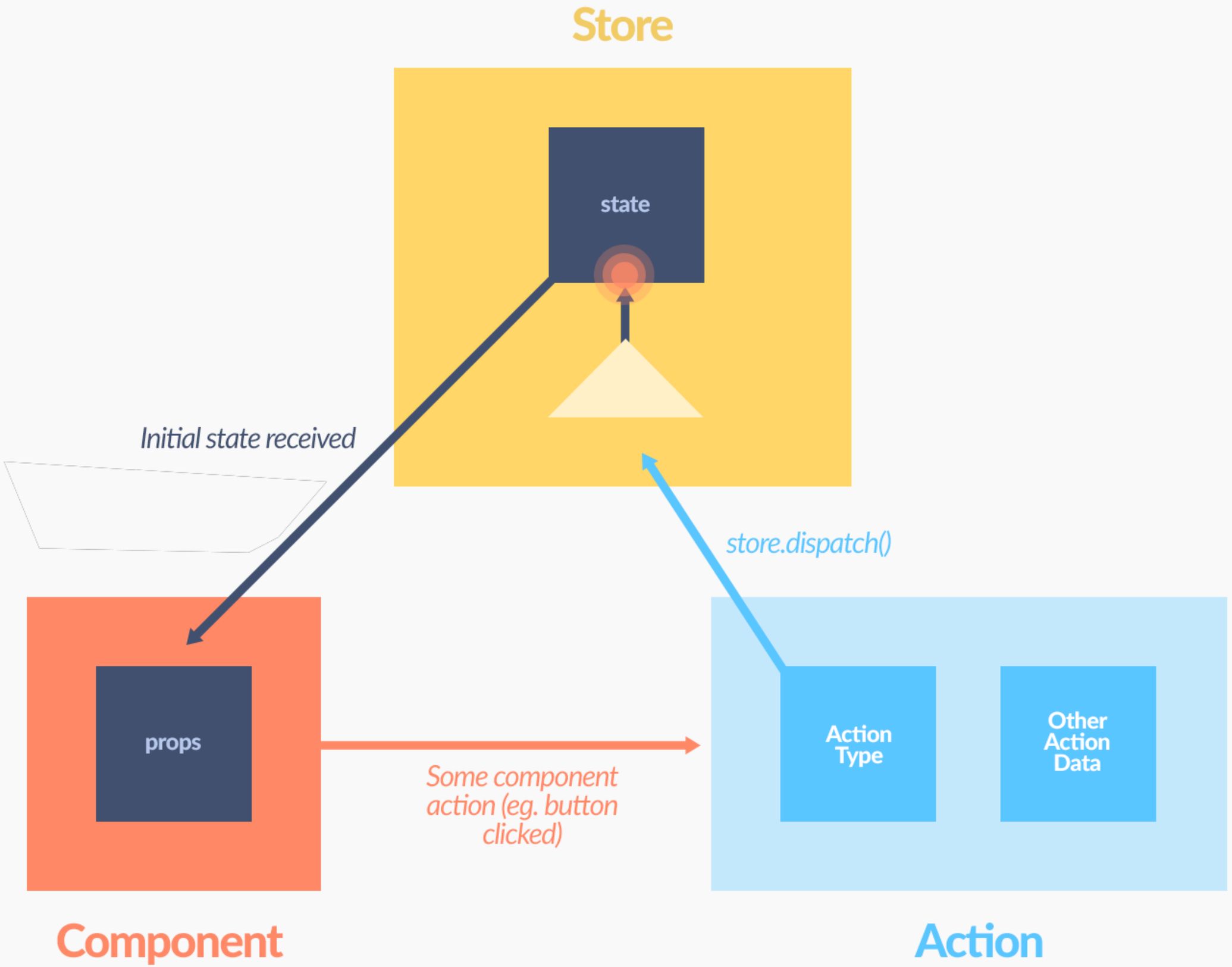
ACTION = INTENT

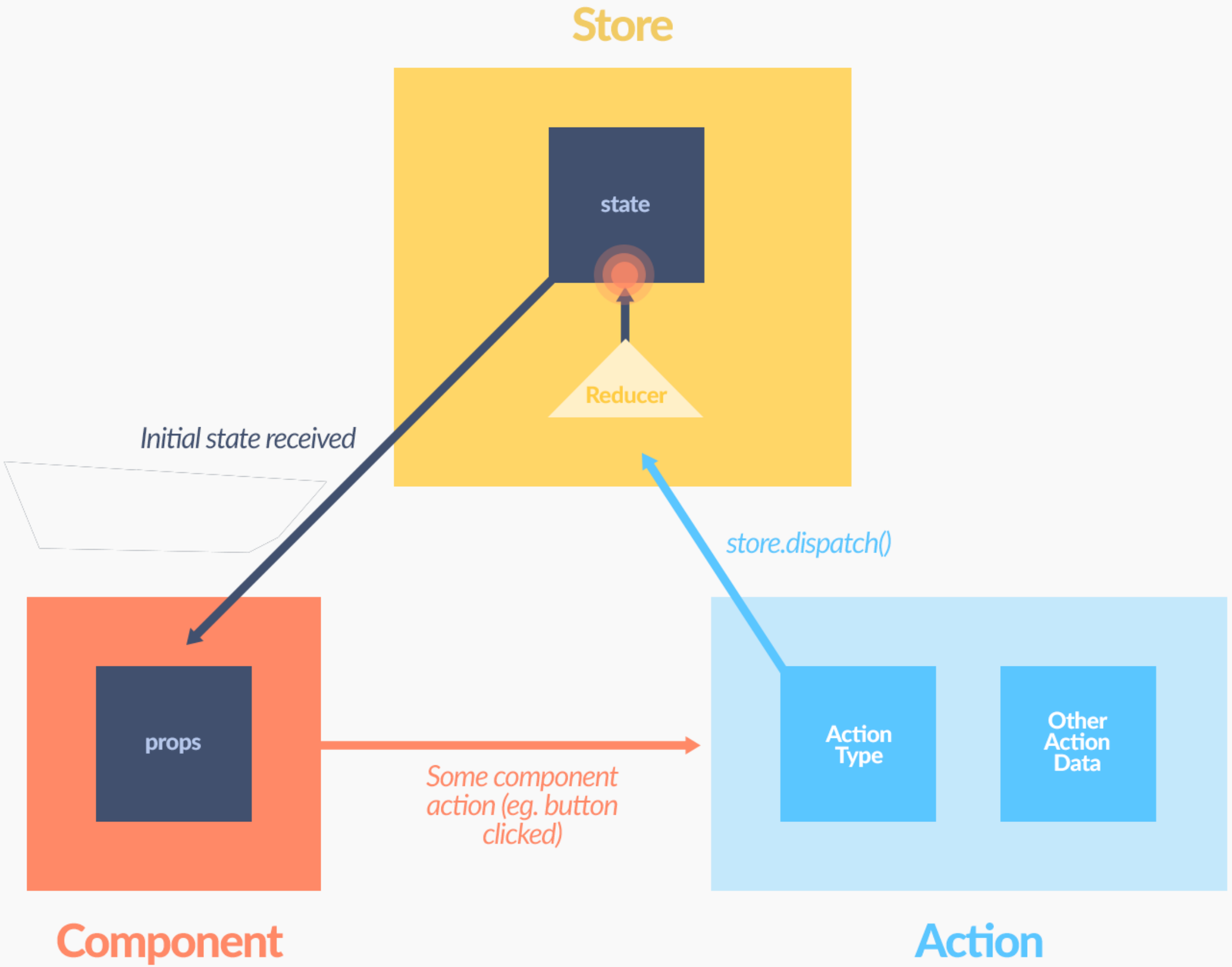
```
{
    // action type
    type: 'USER_CLICKED_BUY_BUTTON',
    // action data
    productId: 1
}
```

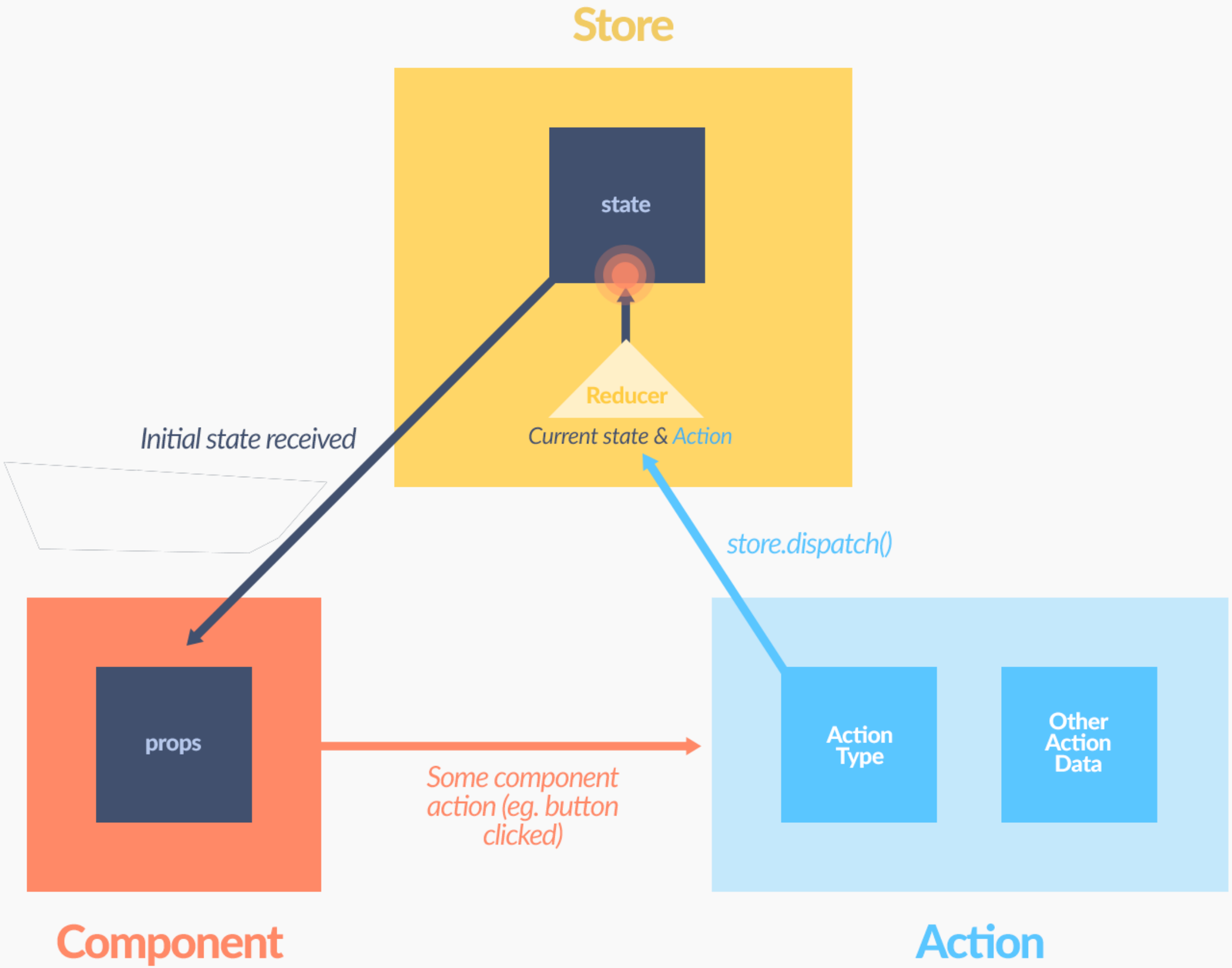
```
store.dispatch({
    type: 'USER_CLICKED_BUY_BUTTON',
    productId: 1
})
```

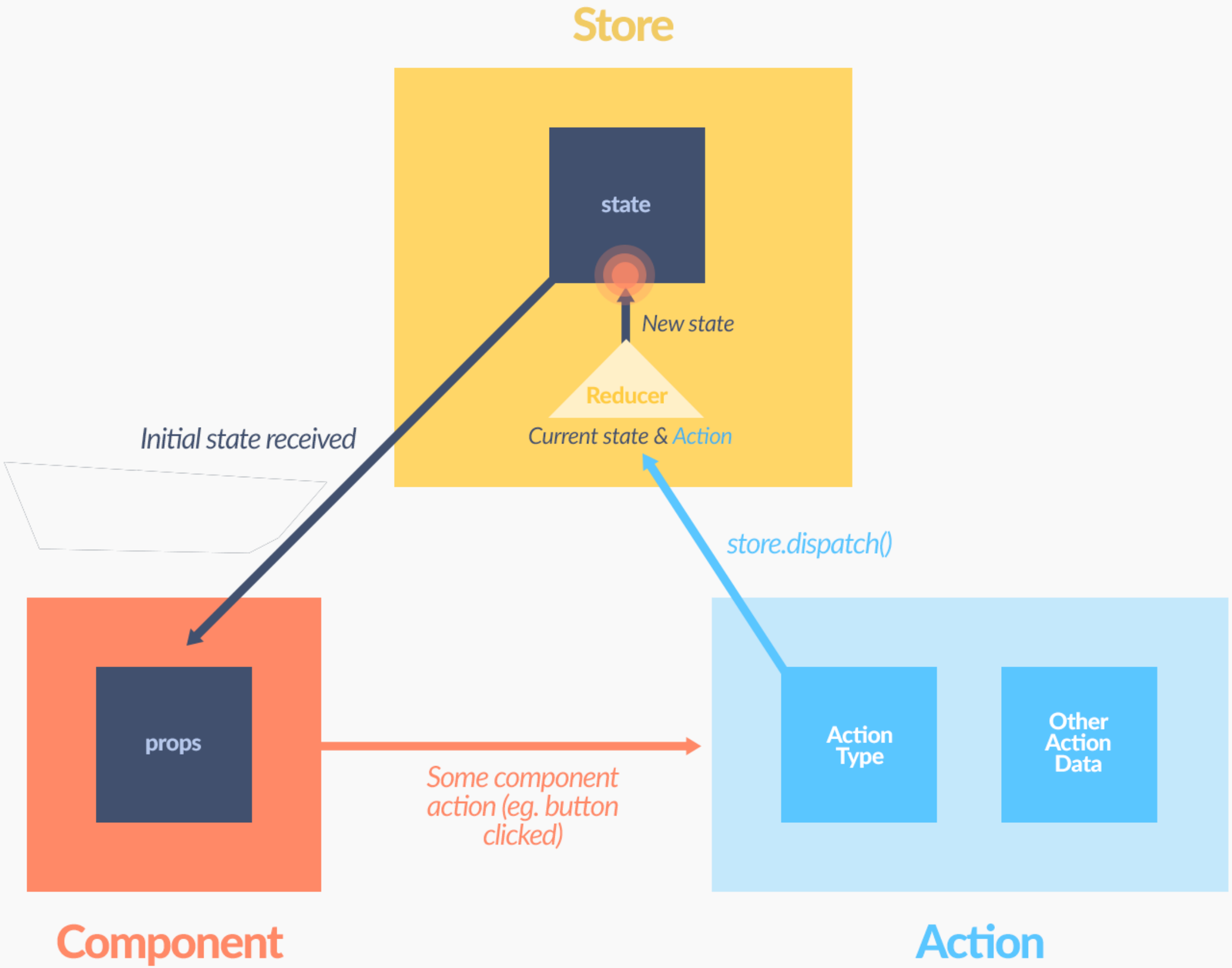
```
function userClickedBuyButton(productId) {
    return {
        type: 'USER_CLICKED_BUY_BUTTON',
        productId
    }
}
```

```
store.dispatch(userClickedBuyButton(1));
```


Store
state
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

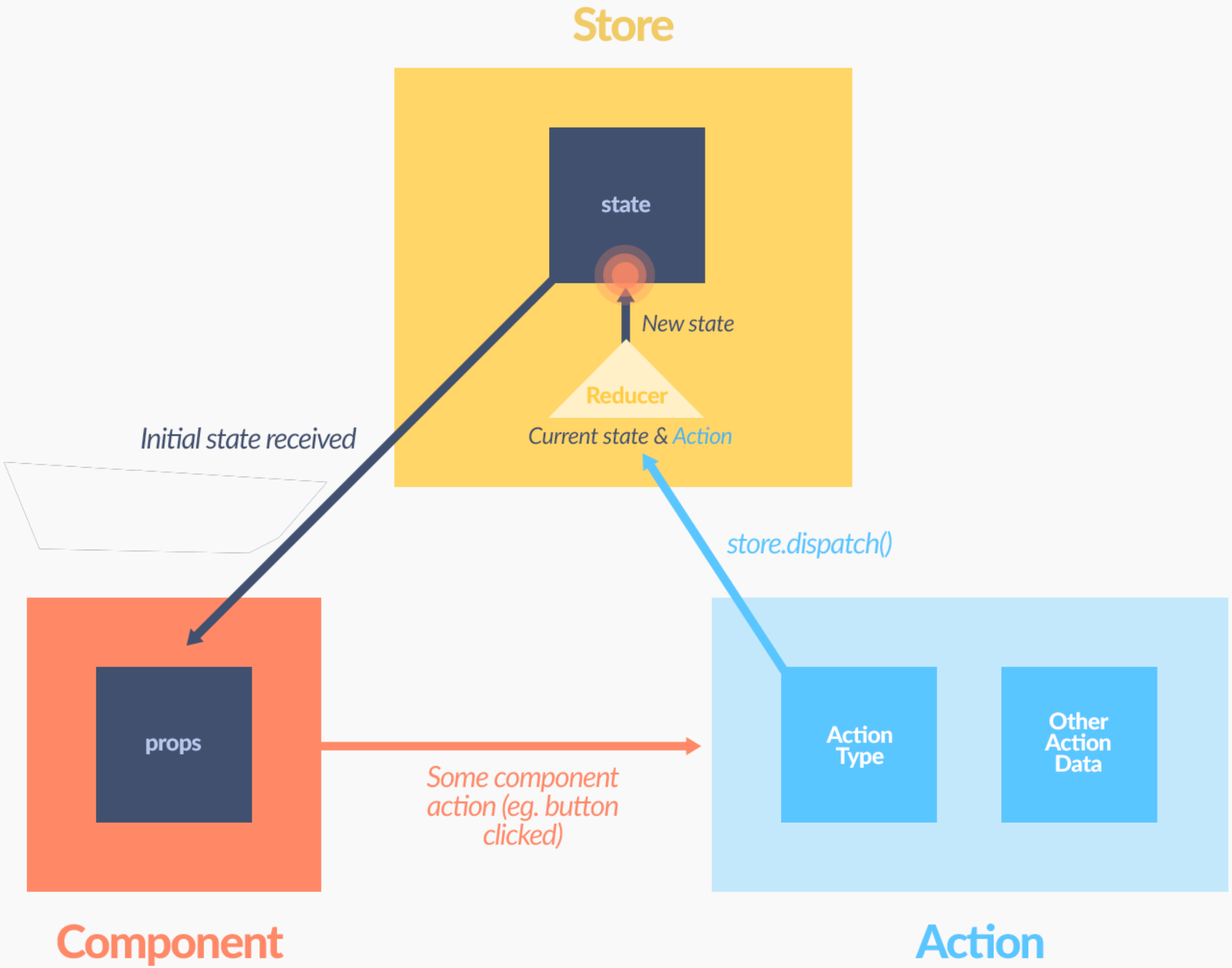

Store
state
Reducer
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

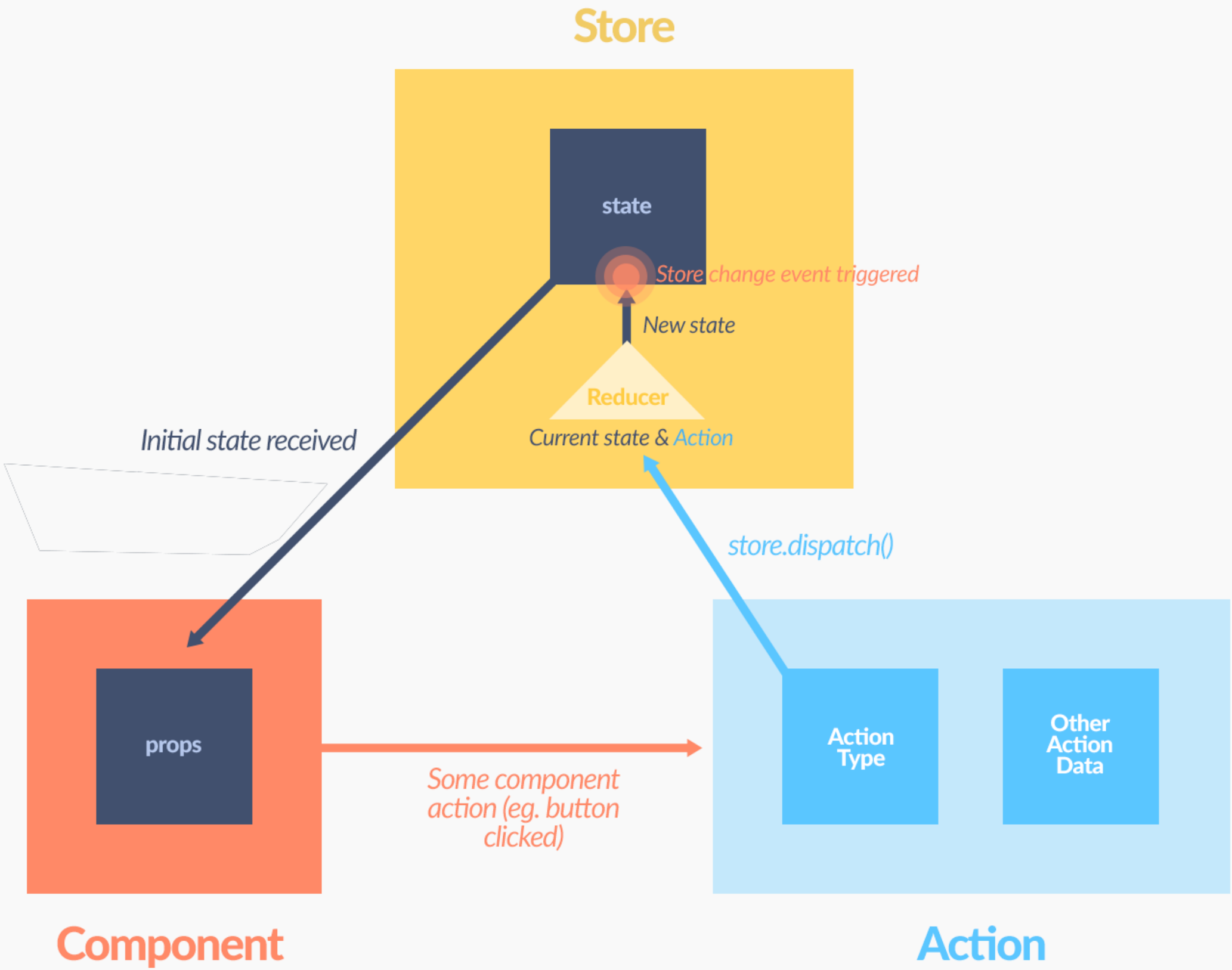

Store
state
Reducer
Initial state received
Current state & Action
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

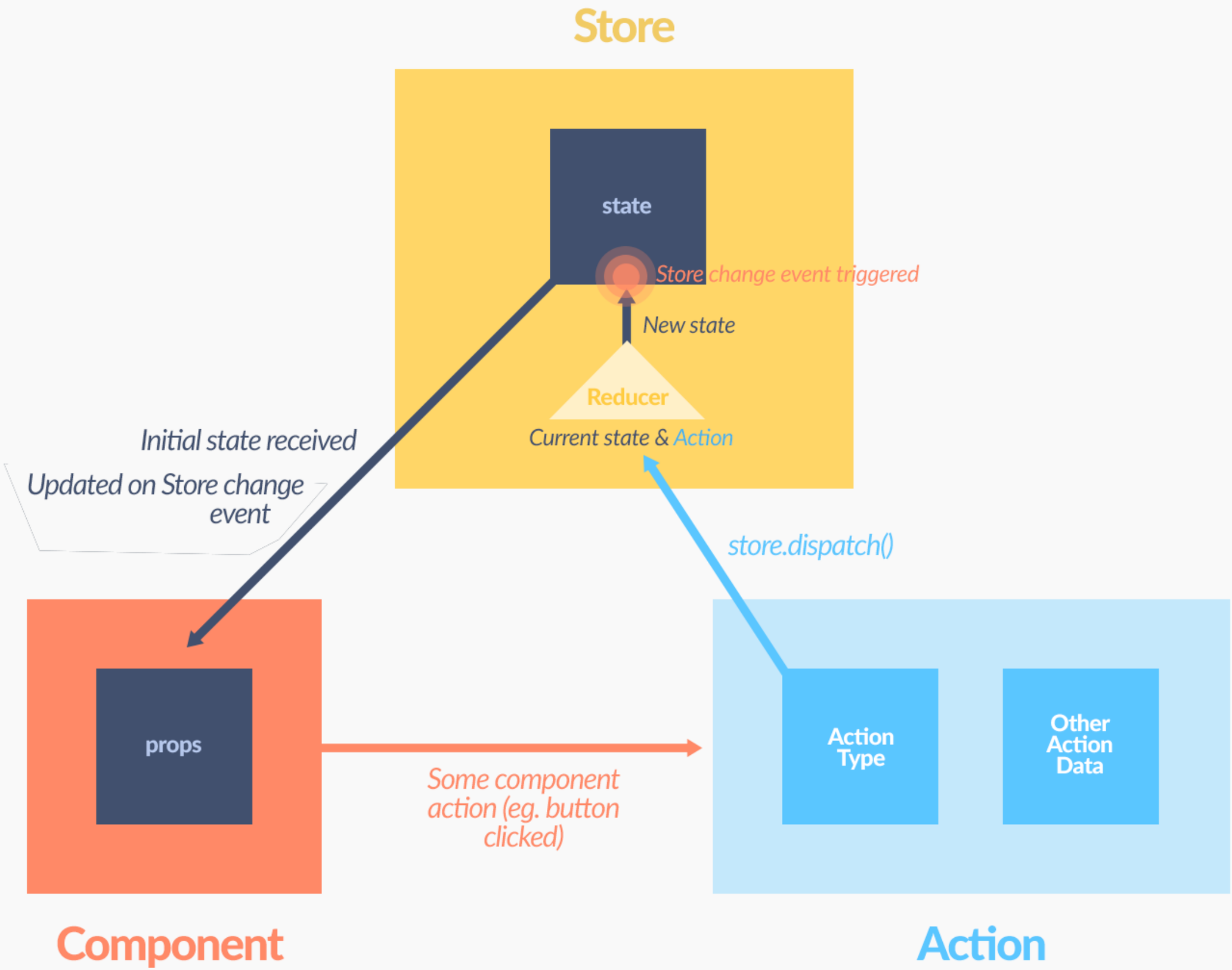

Store
state
New state
Reducer
Current state & Action
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

STATE, ACTION => STATE

```
function reducer(state=0, action){
    if (action.type === 'INCREMENT_CLICKED') {
        return state + 1;
    }
    return state;
}
```


Store
state
New state
Reducer
Current state & Action
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action


Store
state
Store change event triggered
New state
Reducer
Current state & Action
Initial state received
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action


Store
state
Store change event triggered
New state
Reducer
Current state & Action
Initial state received
Updated on Store change event
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action

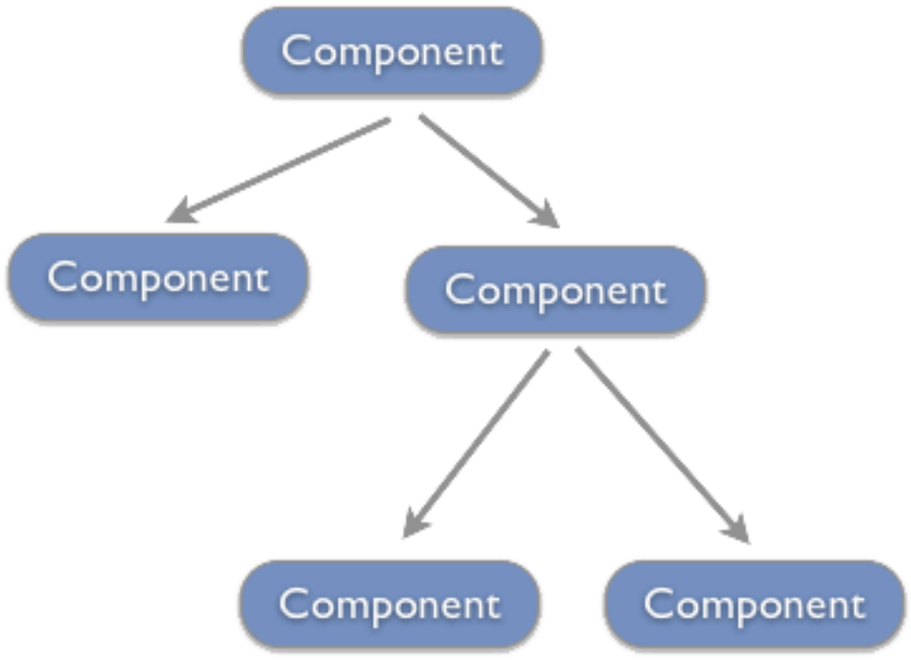
```
import { createStore, combineReducers } from 'redux';

import { connect, Provider } from 'react-redux';
```

```
const store = createStore(reducer);
```

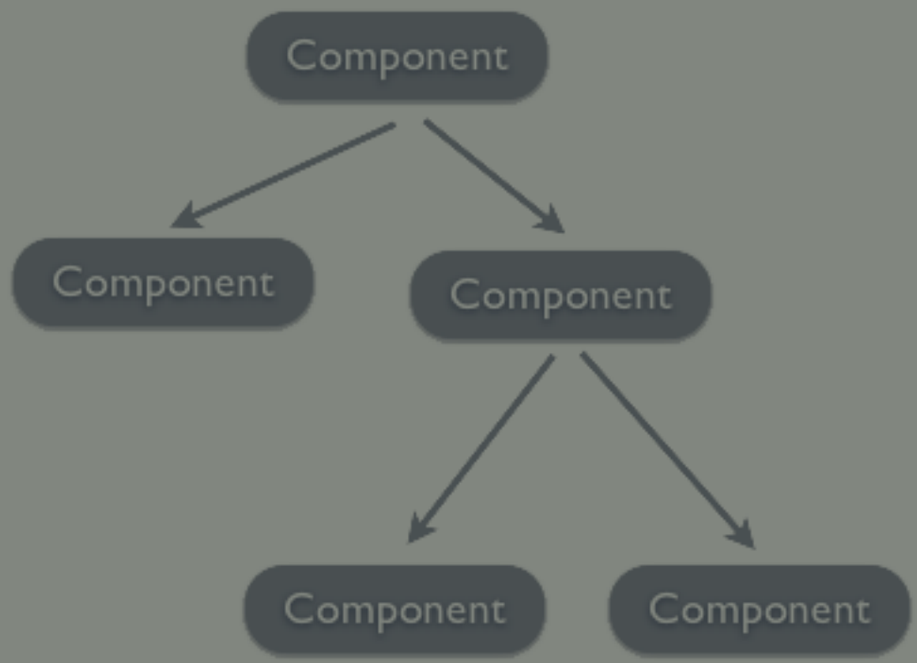
```
function reducer(state, action) {
    ...
    return state;
}

const store = createStore(reducer);
```

```
const initialState = ...

function reducer(state=initialState, action) {
    ...
    return state;
}

const store = createStore(reducer);
```

```
combineReducers({
    tweets: tweetReducer,
    viewState: viewStateReducer,
    route: routeReducer
})
```


Component
Component
Component
Component
Component

```
connect()()

mapStateToProps()
```



```
const mapStateToProps = state => {
    return {
        hasTweets: state.tweets.length > 0,
        mostRecentTweet: state.tweets[0]
    }
}

connect(mapStateToProps)(MyComponent)
```

```
<Provider>
    <App />
</Provider>
```

PRAKTISK DEL

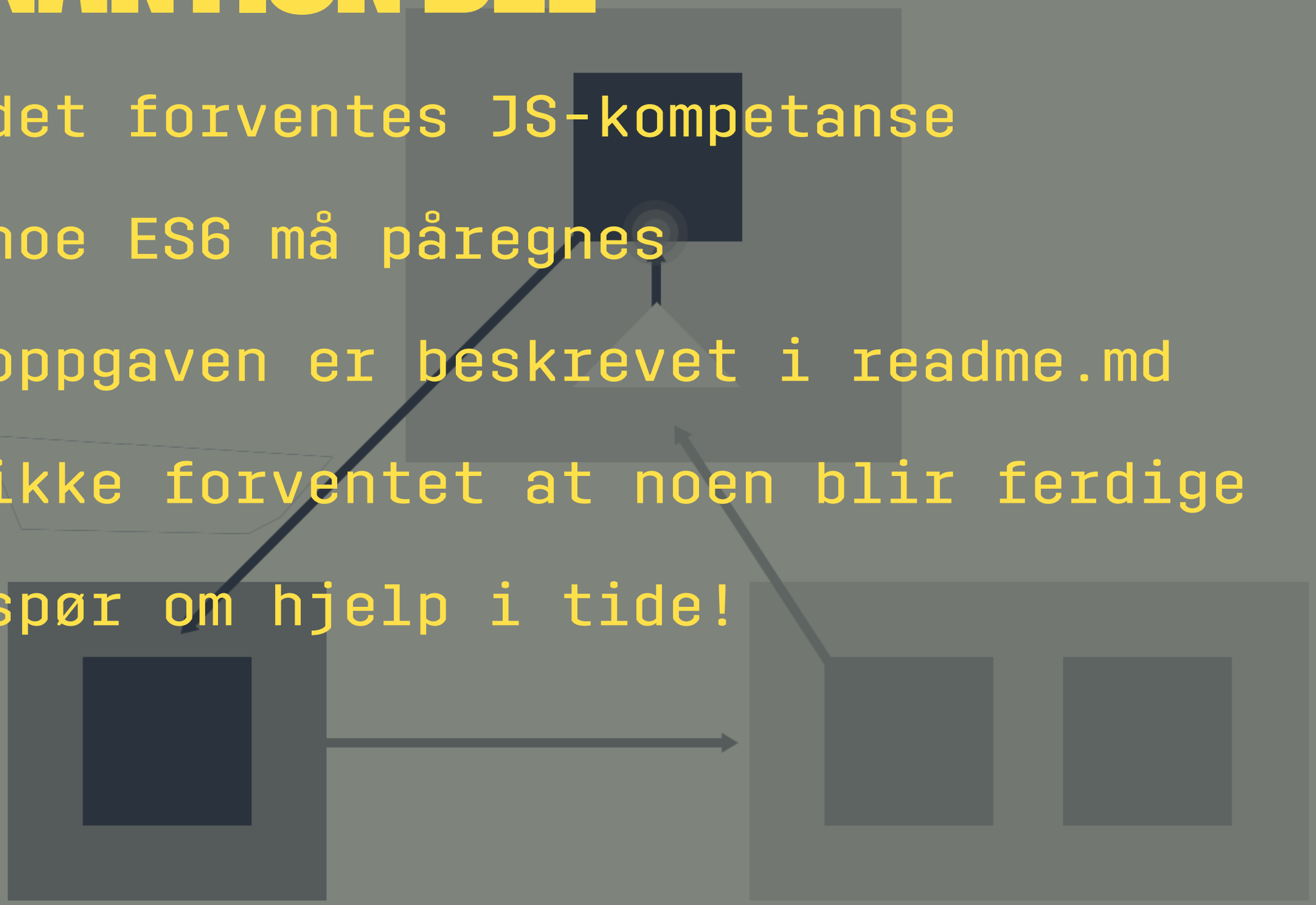
» det forventes JS-kompetanse

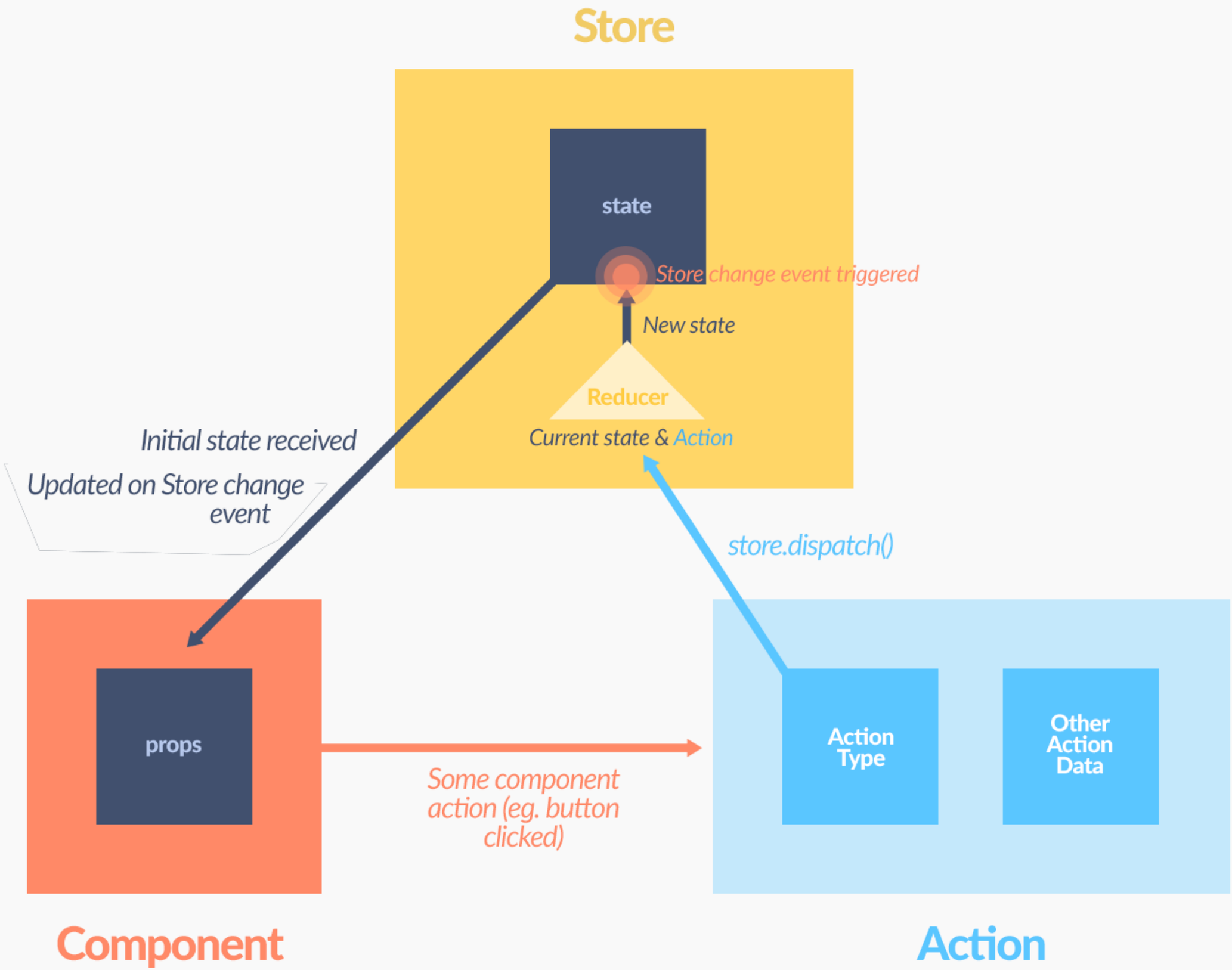
» noe ES6 må påregnes

» oppgaven er beskrevet i readme.md

» ikke forventet at noen blir ferdige

» spør om hjelp i tide!




Store
state
Store change event triggered
New state
Reducer
Current state & Action
Initial state received
Updated on Store change event
store.dispatch()
props
Some component action (eg. button clicked)
Action Type
Other Action Data
Component
Action