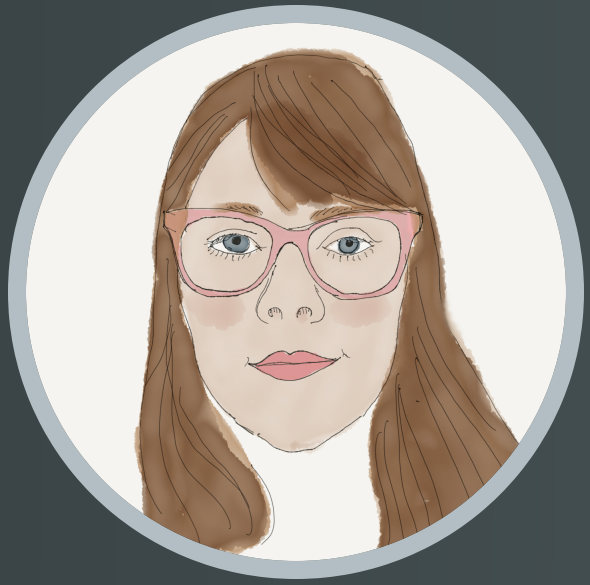
The State of React State in 2019

Becca Bailey • React Conf 2019

@beccaliz

Hi, I'm Becca!



Chicago, IL

@beccaliz


ADVERTISING


Drinks in Queue
2 minutes
MAKR SHAKR
Drinks Re

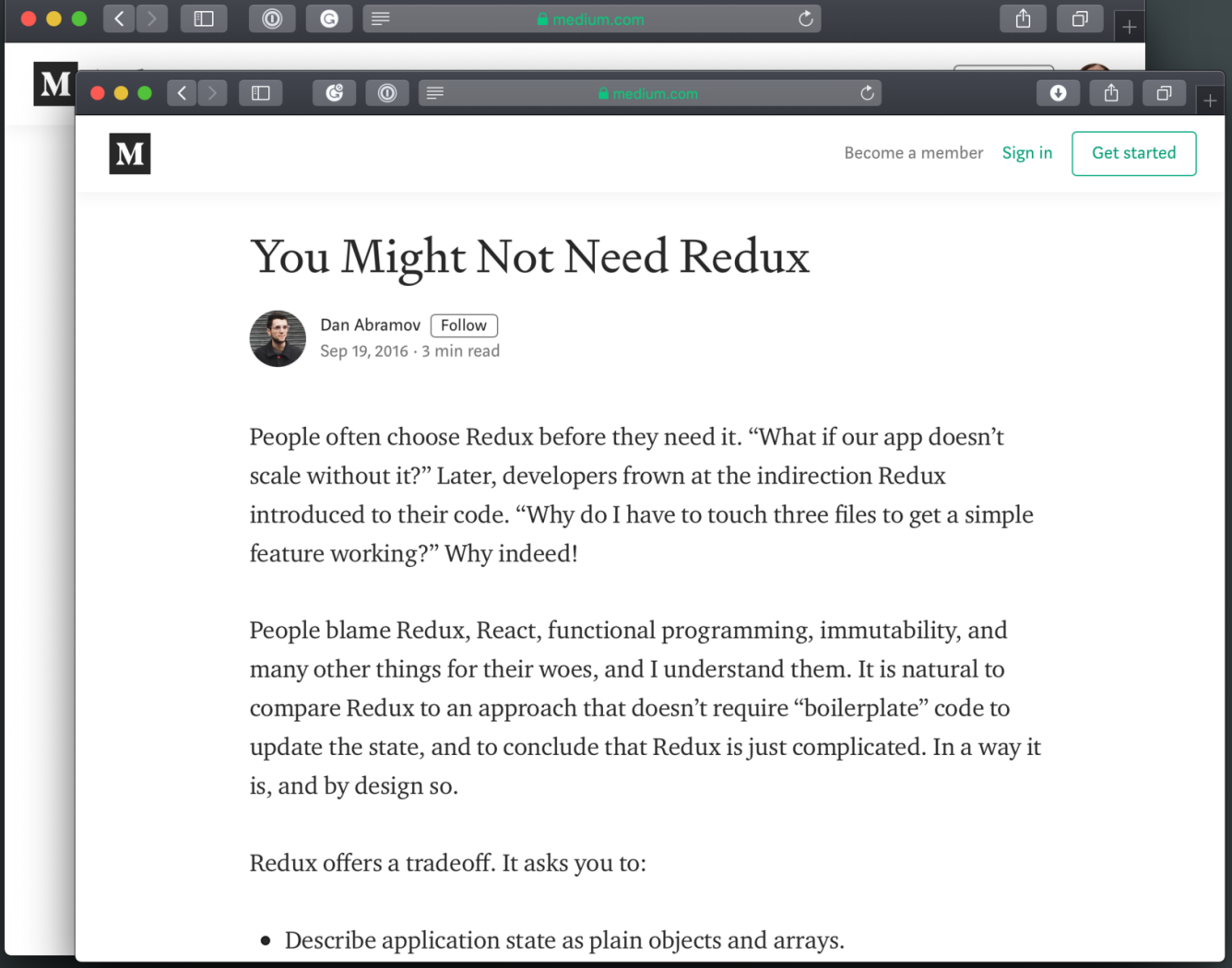
Why this talk?

The Newbie

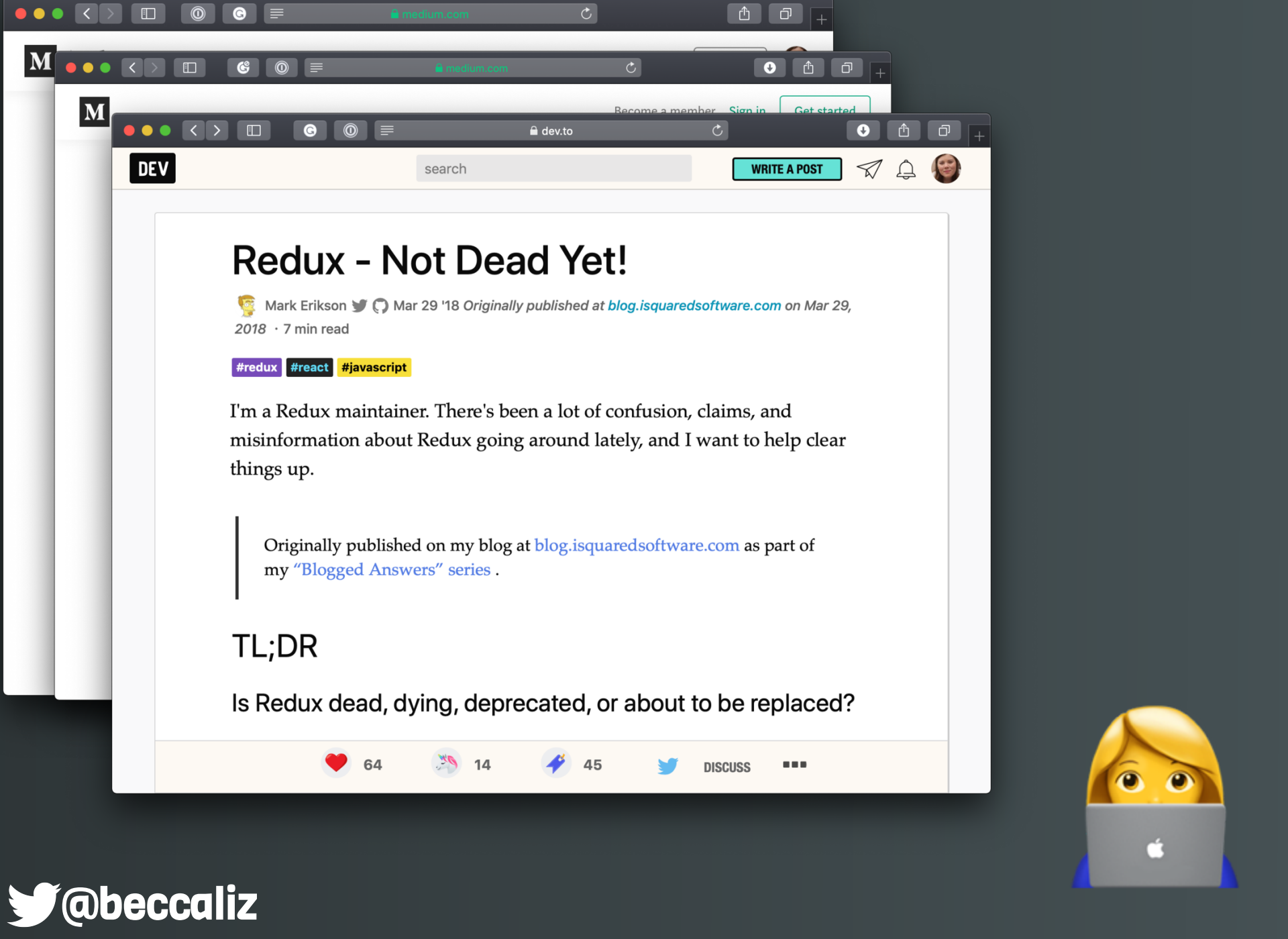
@beccaliz


medium.com
CodingTheSmartWay.com
Upgrade
Learn Redux
Introduction To
State Management
With React
CodingTheSmartWay.com
Learn Redux — Introduction To State Management With React
Sebastian Eschweiler
Follow
May 15, 2017 · 9 min read

@beccaliz


medium.com
Become a member
Sign in
Get started
You Might Not Need Redux
Dan Abramov
Follow
Sep 19, 2016 · 3 min read
People often choose Redux before they need it. “What if our app doesn’t scale without it?” Later, developers frown at the indirection Redux introduced to their code. “Why do I have to touch three files to get a simple feature working?” Why indeed!
People blame Redux, React, functional programming, immutability, and many other things for their woes, and I understand them. It is natural to compare Redux to an approach that doesn’t require “boilerplate” code to update the state, and to conclude that Redux is just complicated. In a way it is, and by design so.
Redux offers a tradeoff. It asks you to:
Describe application state as plain objects and arrays.

@beccaliz


medium.com
Become a member
Sign in
Get started
dev.to
DEV
search
WRITE A POST
Redux - Not Dead Yet!
Mark Erikson Mar 29 '18 Originally published at blog.isquaredsoftware.com on Mar 29, 2018 · 7 min read
#redux #react #javascript
I'm a Redux maintainer. There's been a lot of confusion, claims, and misinformation about Redux going around lately, and I want to help clear things up.
Originally published on my blog at blog.isquaredsoftware.com as part of my "Blogged Answers" series .
TL;DR
Is Redux dead, dying, deprecated, or about to be replaced?
64
14
45
DISCUSS

@beccaliz

@beccaliz

The Loyalist

"If it ain't broke,
don't fix it!"

@beccaliz

@beccaliz

redux

react

@beccaliz

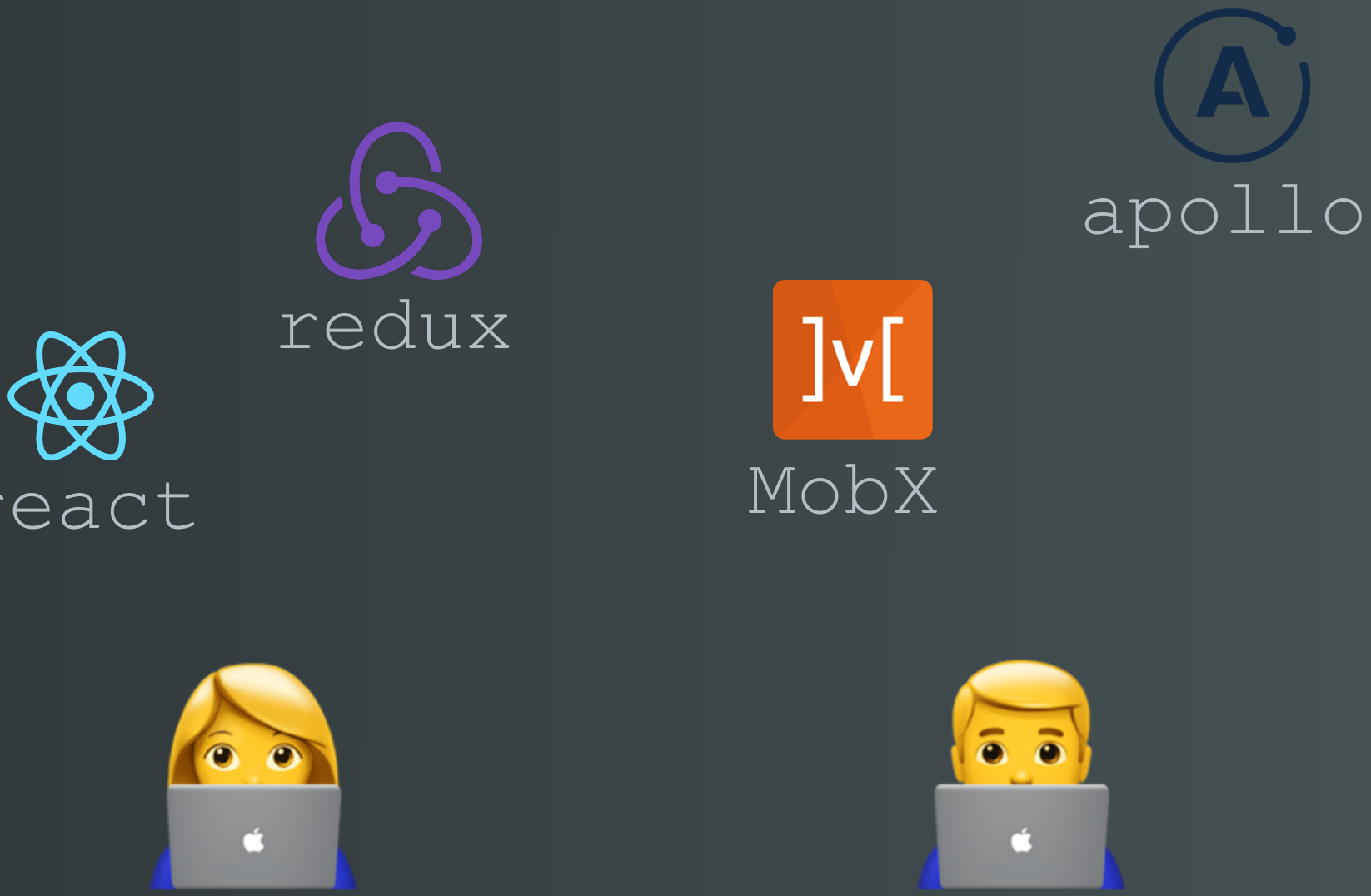

redux


react


redux


react


MobX



@beccaliz

context

redux

react

MobX

context

apollo

redux

react

MobX

hooks

@beccaliz

context

redux

react

MobX

hooks

XSTATE

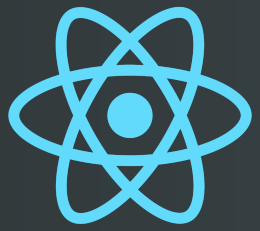
@beccaliz

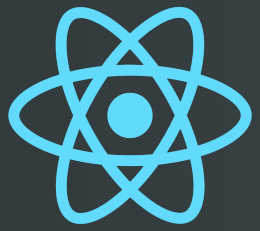
@beccaliz

The Explorer

@beccaliz

@beccaliz

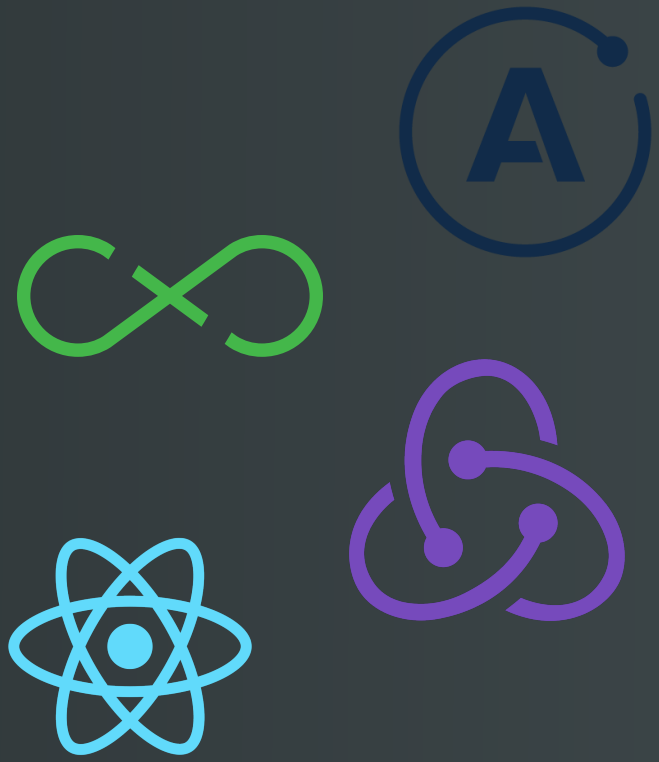




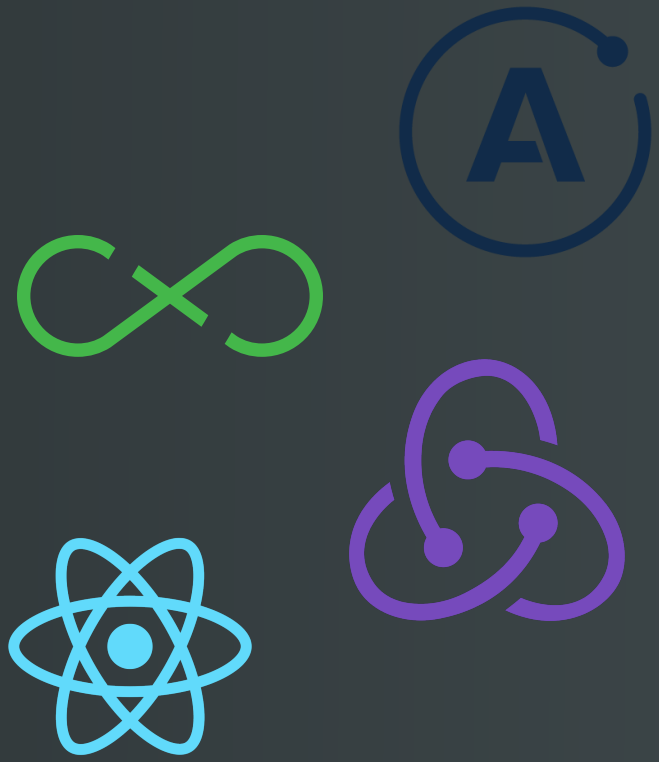
@beccaliz

@beccaliz

@beccaliz

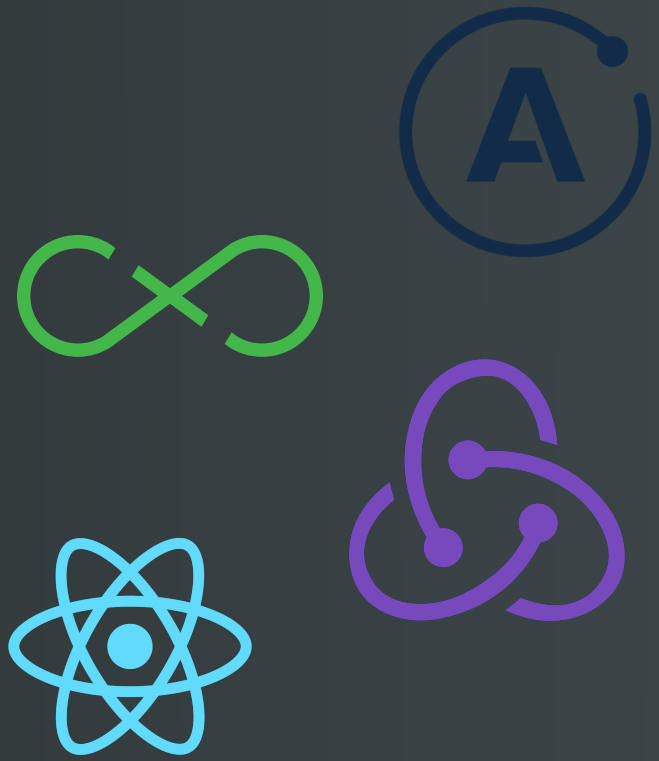


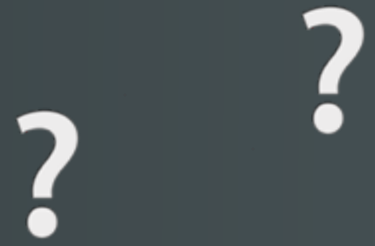
@beccaliz



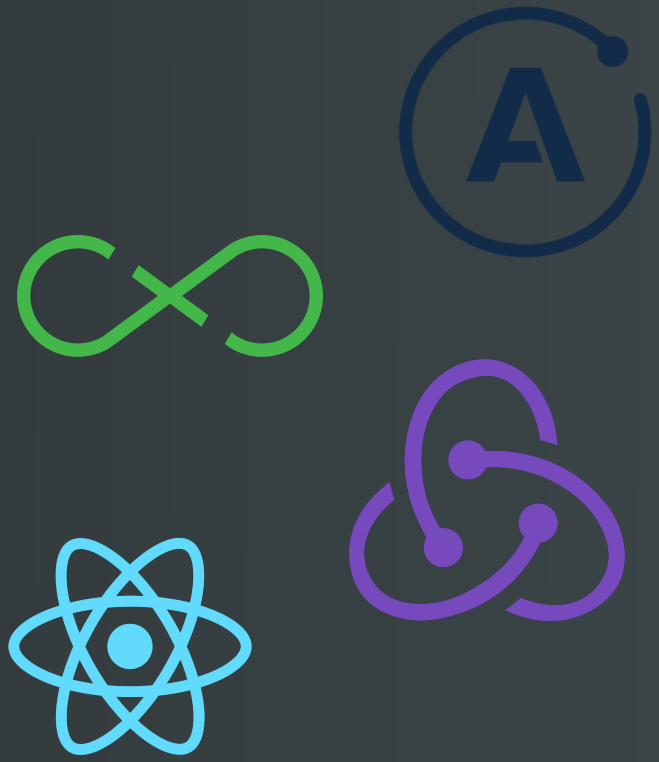
?

@beccaliz



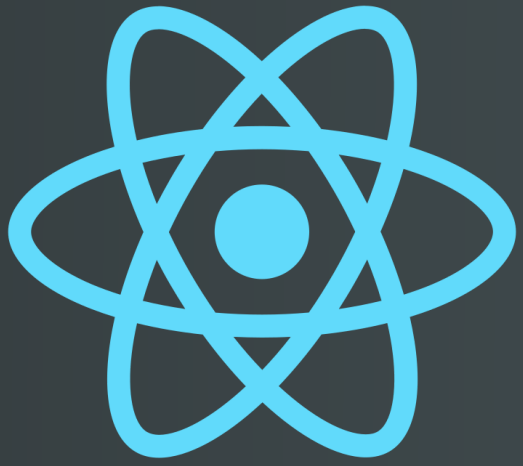


@beccaliz

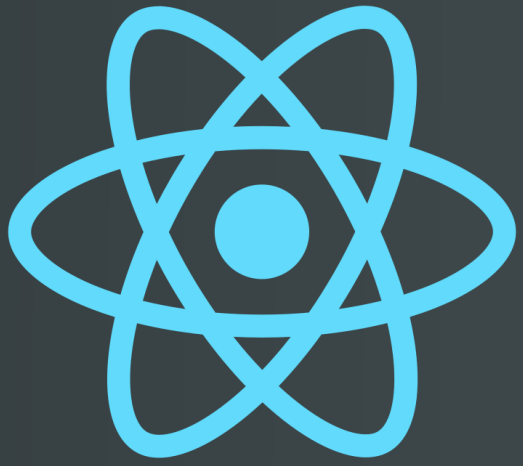


@beccaliz

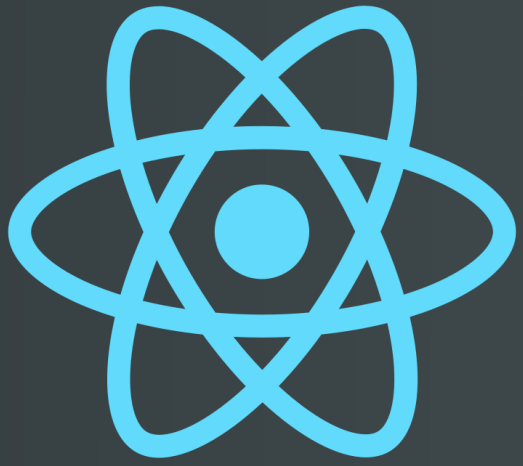
@beccaliz



@beccaliz

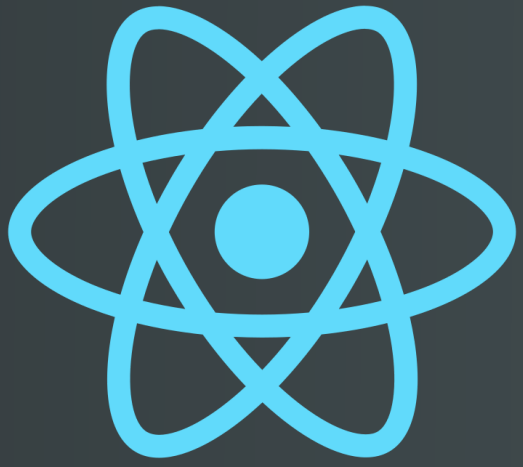


@beccaliz



@beccaliz



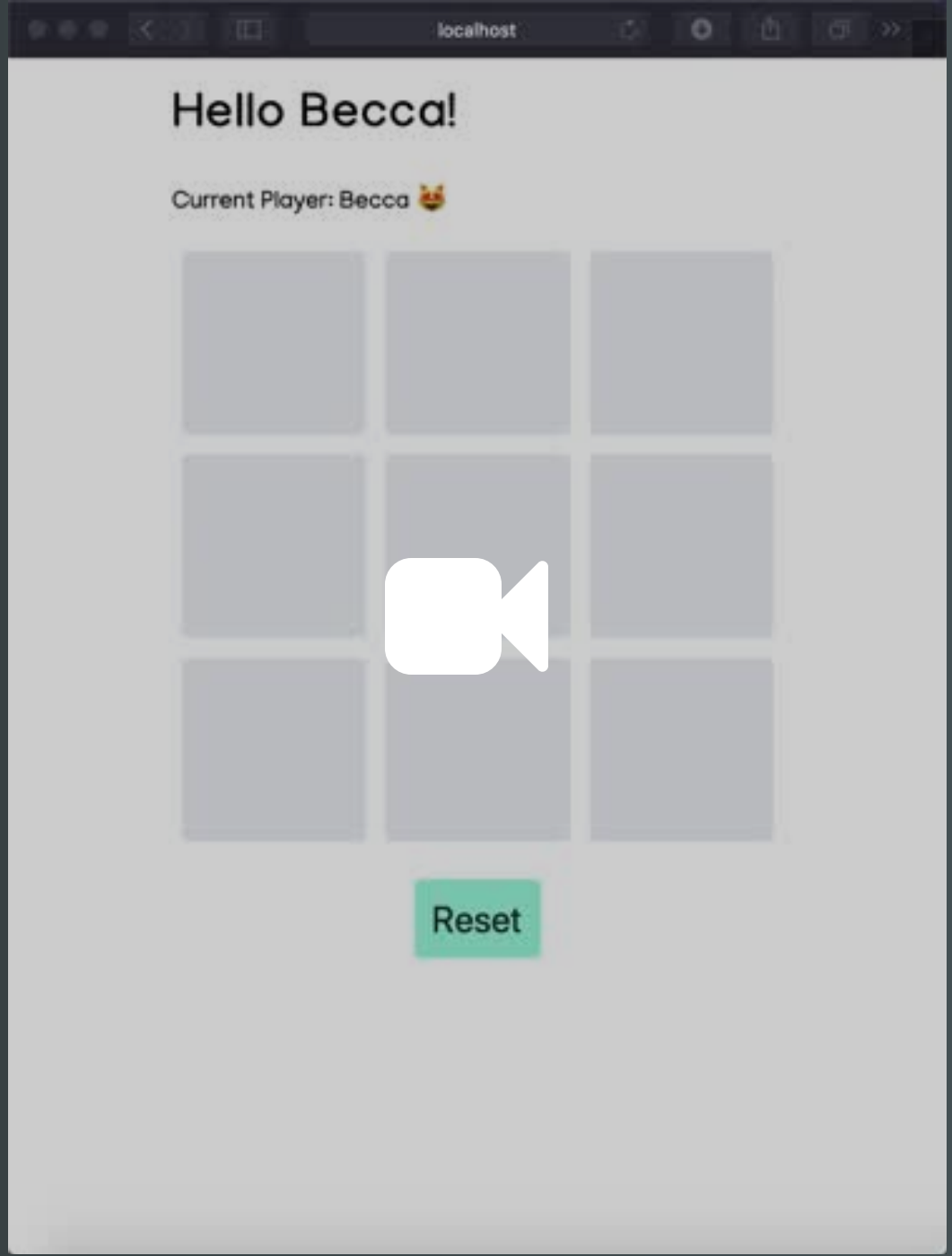

@beccaliz

@beccaliz

1. Clarify the problem

1. Clarify the problem
2. Explore some solutions

1. Clarify the problem
2. Explore some solutions
3. Make decisions


localhost
Hello Becca!
Current Player: Becca
Reset

State

@beccaliz

setState

setState

useState

Local State

@beccaliz

You win!

Yay!

```
class MyComponent extends React.Component {
  state = {
    visible: false
  };

  showModal() {
    this.setState(state => ({
      visible: true
    }));
  }

  hideModal() {
    this.setState(state => ({
      visible: false
    }));
  }

  render() {
    ...stuff here
  }
}
```

```
class MyComponent extends React.Component {
  state = {
    visible: false
  };

  showModal() {
    this.setState(state => ({
      visible: true
    }));
  }

  hideModal() {
    this.setState(state => ({
      visible: false
    }));
  }

  render() {
    ...stuff here
  }
}
```

```
class MyComponent extends React.Component {
  state = {
    visible: false
  };

  showModal() {
    this.setState(state => ({
      visible: true
    }));
  }

  hideModal() {
    this.setState(state => ({
      visible: false
    }));
  }

  render() {
    ...stuff here
  }
}
```

```
const MyComponent = () => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true);
  }

  function hideModal() {
    setVisible(false);
  }

  return (
    ...stuff here
  );
};
```

```
const MyComponent = () => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true);
  }

  function hideModal() {
    setVisible(false);
  }

  return (
    ...stuff here
  );
};
```

```
const MyComponent = () => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true);
  }

  function hideModal() {
    setVisible(false);
  }

  return (
    ...stuff here
  );
};
```

Prop Drilling

```
1 // App.js
2 const App = () => {
3
4   return (
5     <Container>
6       <Game></Game>
7     </Container>
8   );
9 };
```

```
const user = {
  id: 123,
  firstName: "Becca",
  lastName: "Bailey",
  email: "beccanelsonbailey@gmail.com",
  marker: "👩‍💻"
}
```

```
// App.js
const App = () => {
  const [user, updateUser] = React.useState();

  React.useEffect(async () => {
    const user = await fetchLoggedInUser();
    updateUser(user);
  }, [])

  return (
    <Container>
      <Game user={user}></Game>
    </Container>
  );
};
```

```
1  // Game.js
2  const Game = ({ user }) => {
3    const [board, updateBoard] = React.useState(EMPTY_BOARD
4
5    function makeMove(index) {
6      updateBoard({...board, [index]: user.marker })
7    }
8
9    return (
10     <React.Fragment>
11       <h1>Hello {user.name}!</h1>
12       <Board board={board} makeMove={makeMove} />
13     </React.Fragment>
14   );
15 };
```

```
1  // Game.js
2  const Game = ({ user }) => {
3    const [board, updateBoard] = React.useState(EMPTY_BOARD
4
5    function makeMove(index) {
6      updateBoard({...board, [index]: user.marker })
7    }
8
9    return (
10     <React.Fragment>
11       <Greeting user={user} />
12       <Board board={board} makeMove={makeMove} />
13     </React.Fragment>
14   );
15 };
```

Repetition

You win!

Yay!

⚠ Error!

Oh no!

showModal

modalOpen

modalIsVisible

modalIsOpen

modalVisible

@beccaliz

@beccaliz

You win!

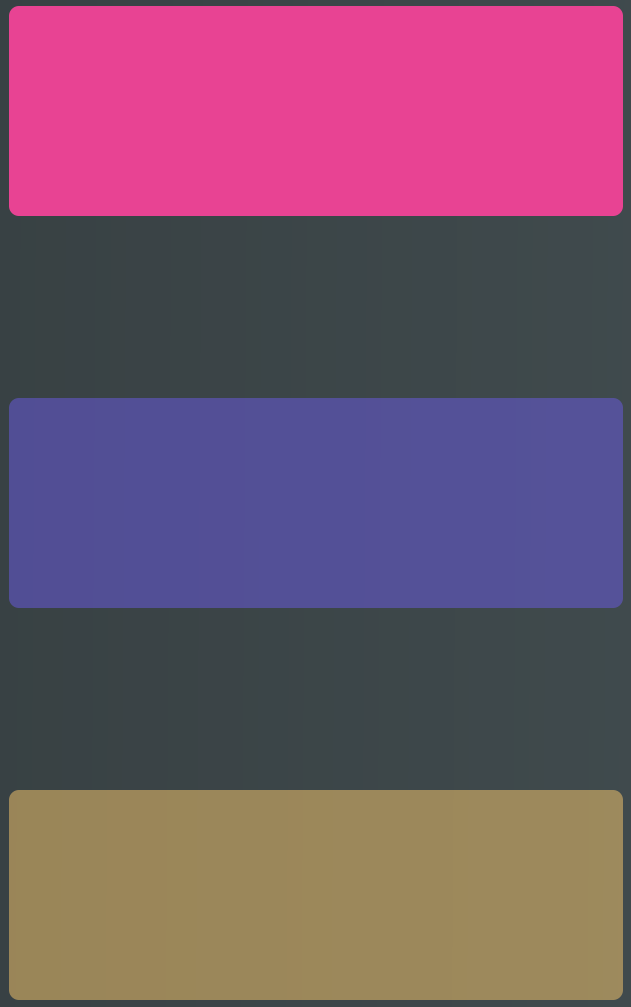
Yay!

⚠ Error!

Oh no!

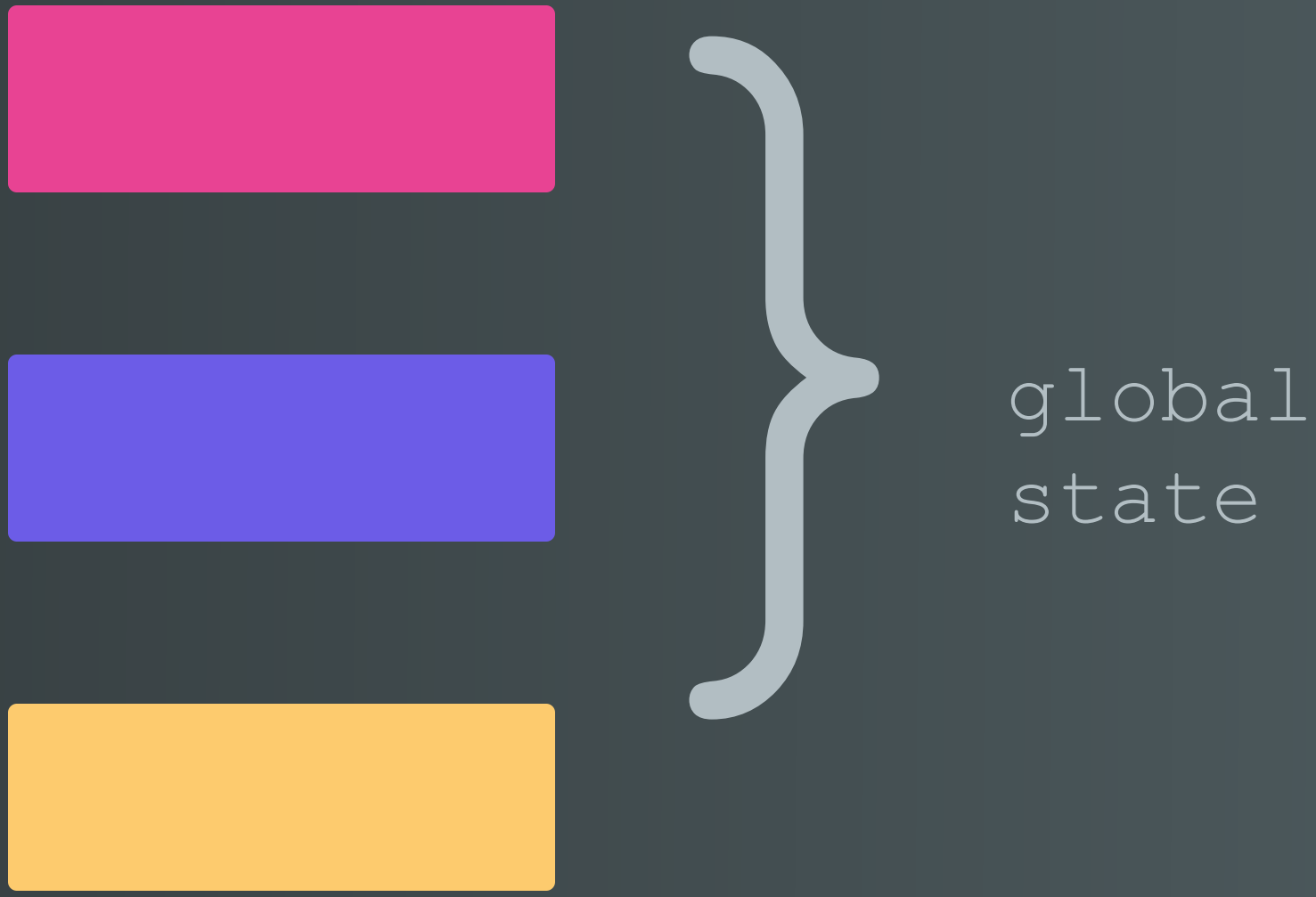
@beccaliz

@beccaliz


} local state

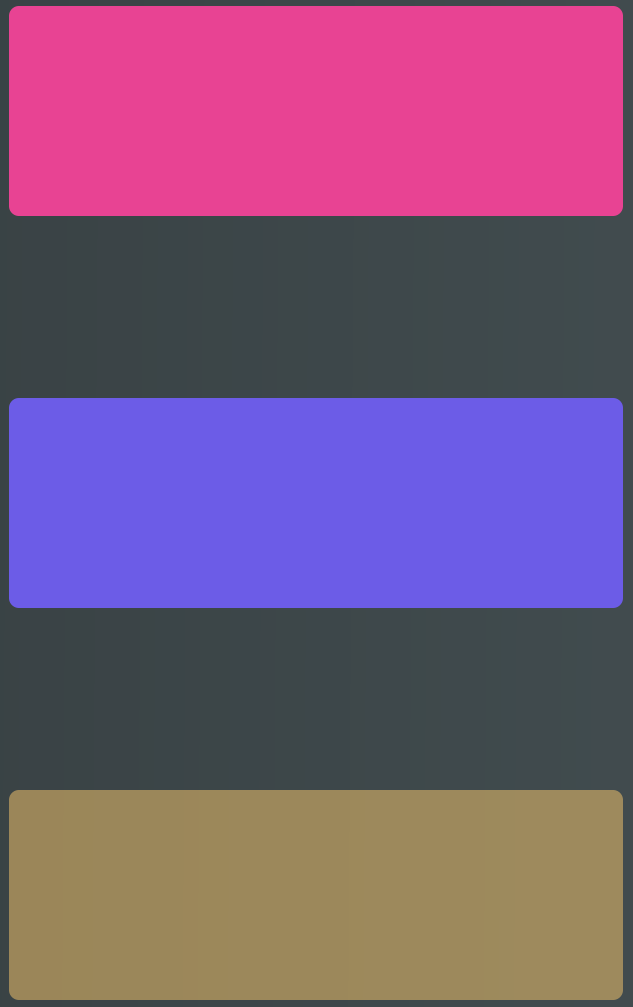
Global state

@beccaliz


global
state

@beccaliz

@beccaliz


semi-local
state

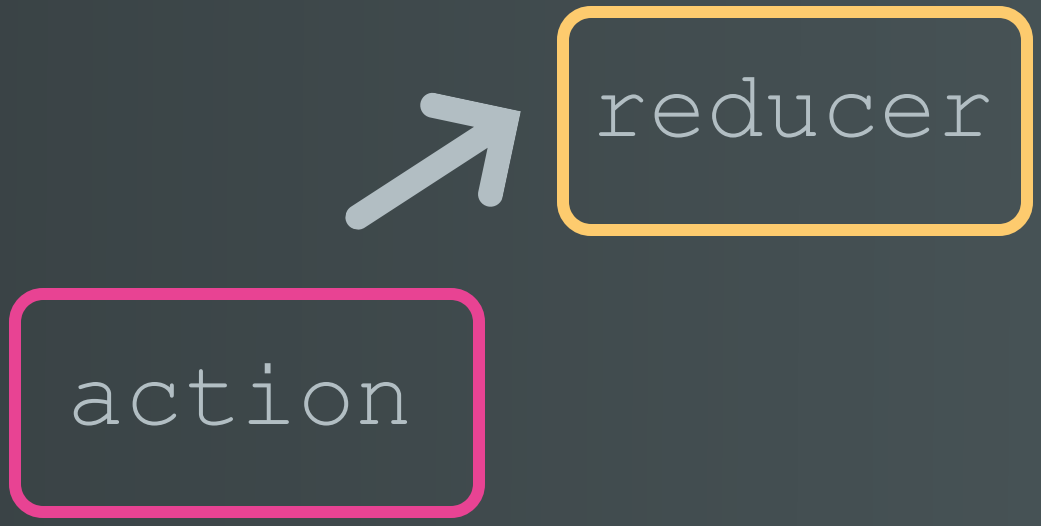
Flux Architecture

@beccaliz

@beccaliz

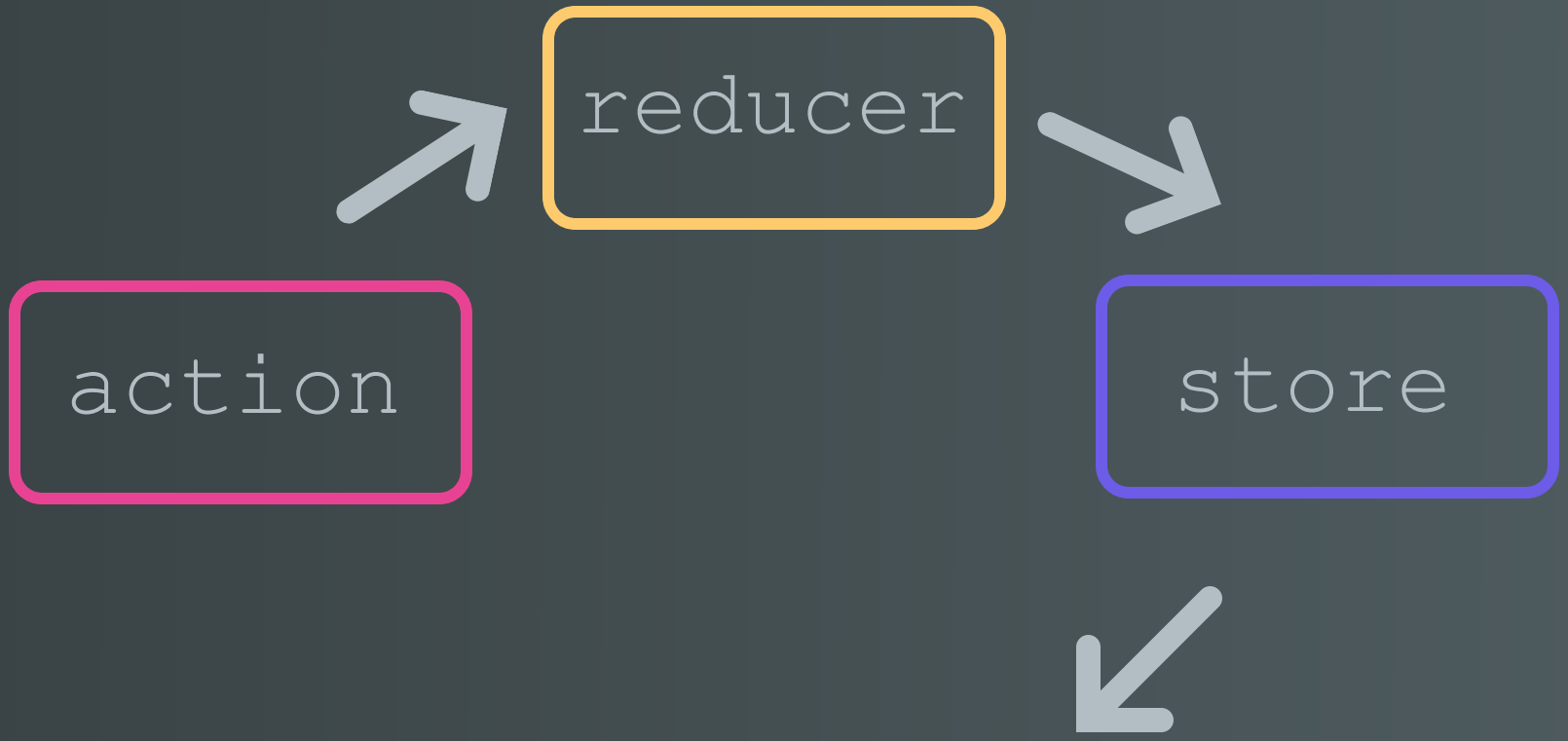
action

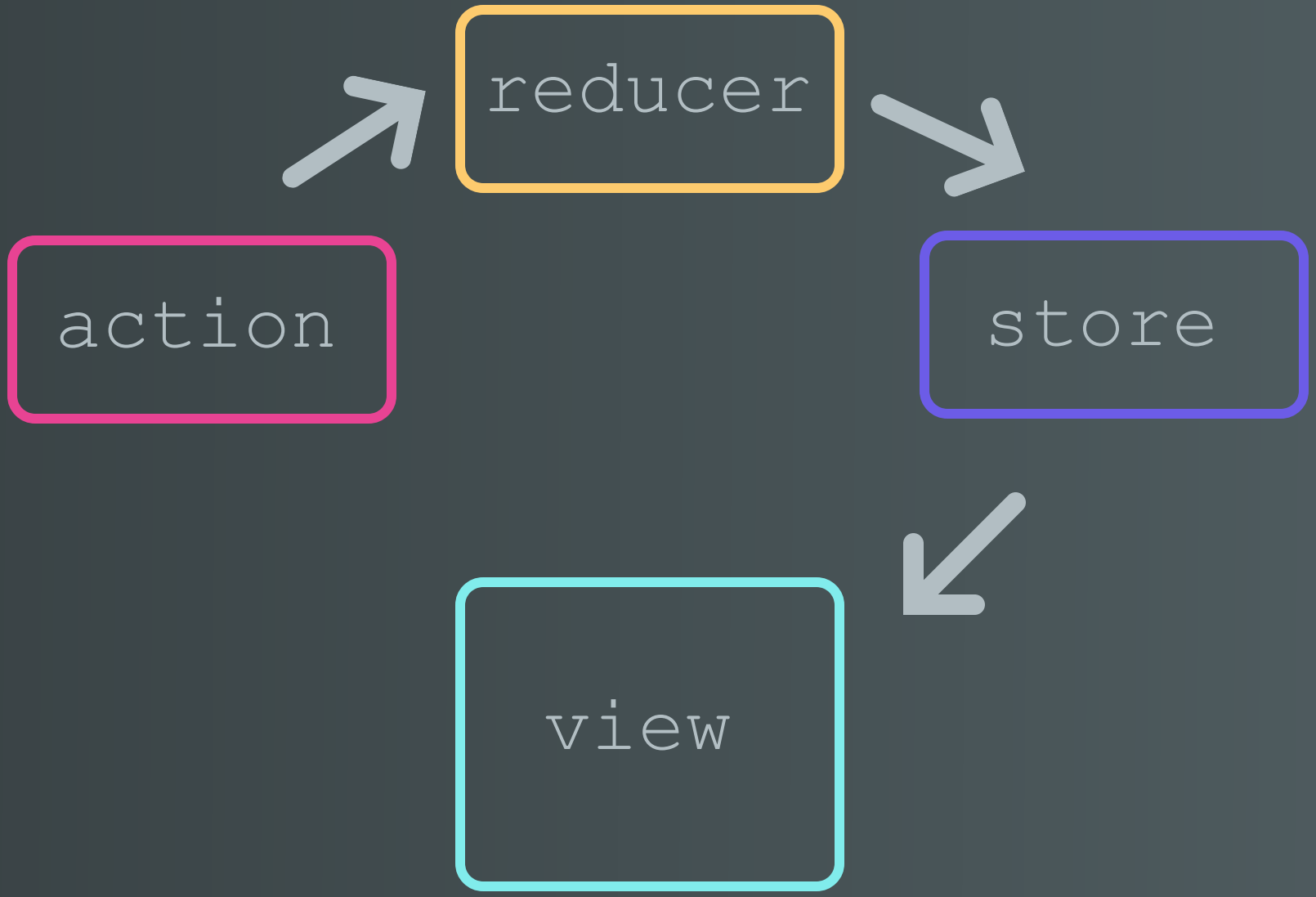

action

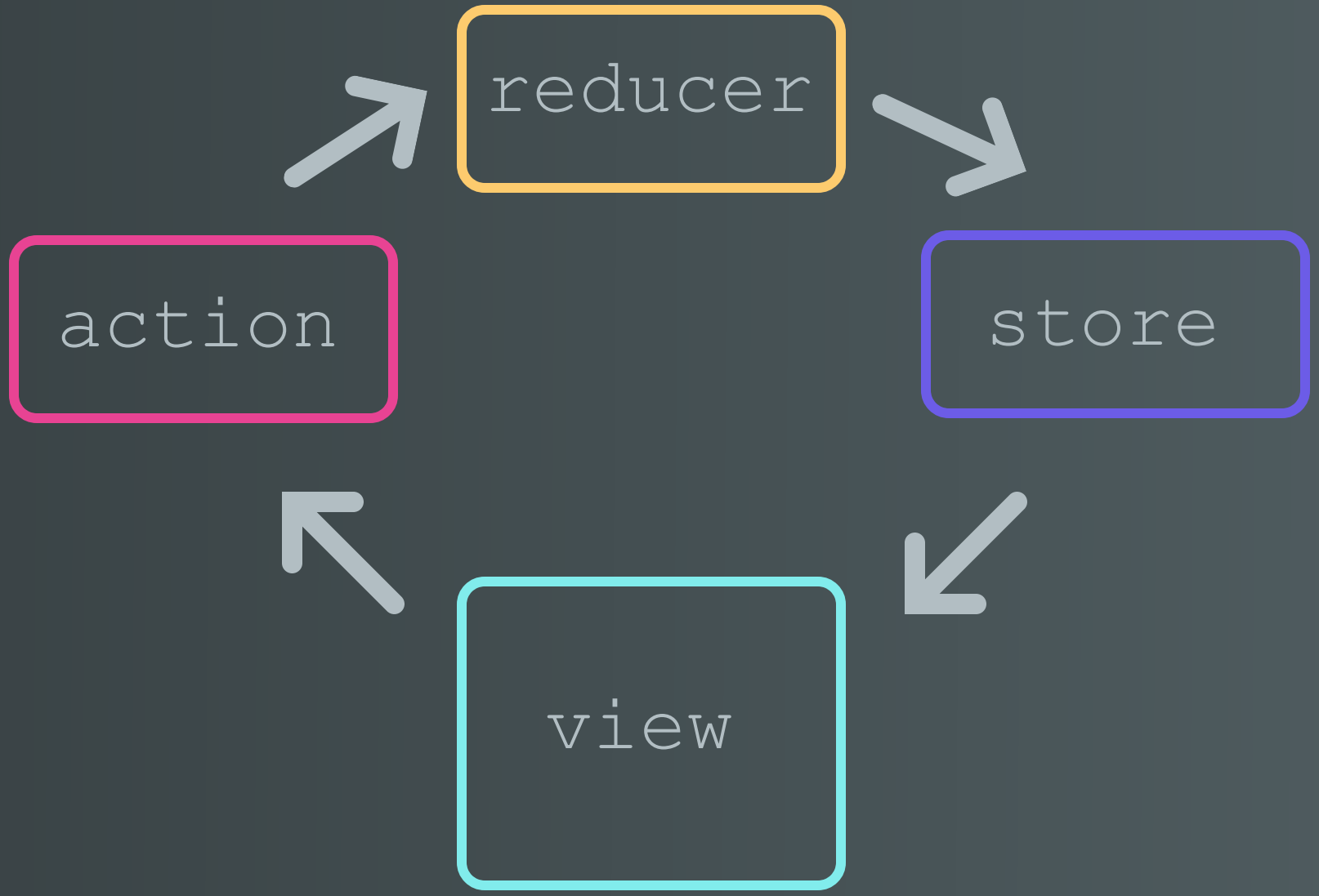

reducer
action


reducer
action


reducer
action
store


reducer
action
store


reducer
action
store
view



@beccaliz

```
// actions.js
export function makeMove(index) {
  return {
    type: "MAKE_MOVE",
    payload: { index }
  };
}

// reducers.js
const game = (state = getInitialState(), action) => {
  switch (action.type) {
    case "MAKE_MOVE": {
      const { index } = action.payload;
      return {
        ...state,
        board: {
          ...state.board,
          [index]: state.currentPlayer.marker,
        }
      };
    }

    default:
      return state;
  }
```

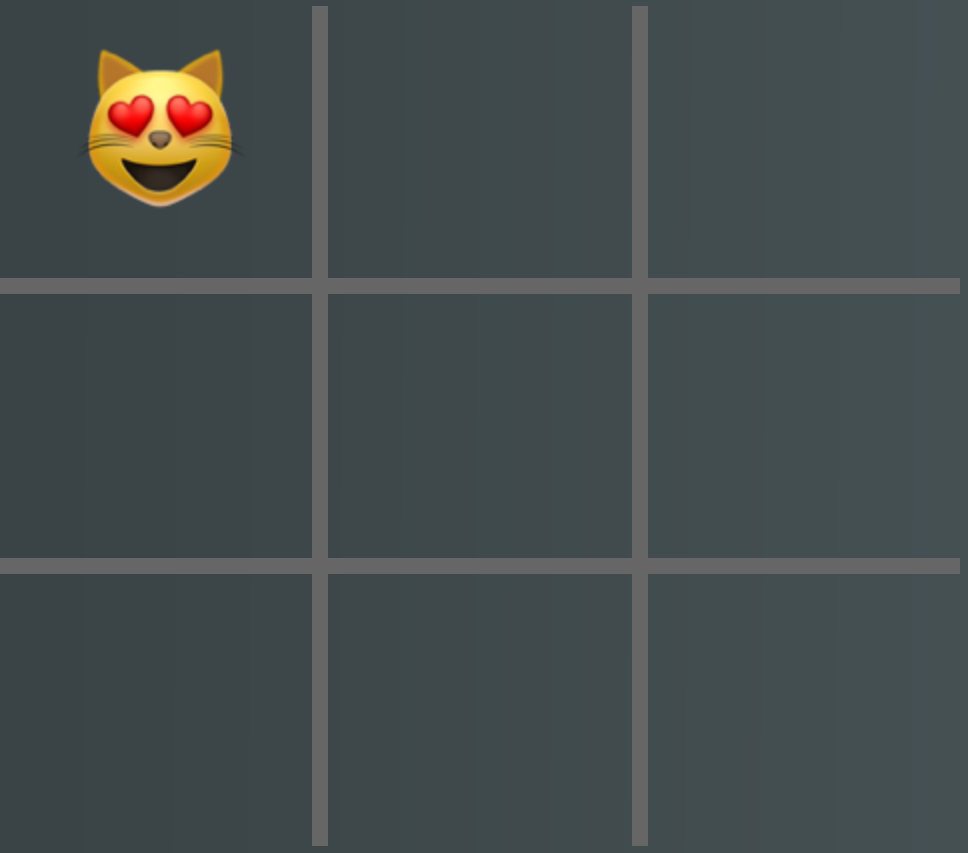
```
// actions.js
export function makeMove(index) {
  return {
    type: "MAKE_MOVE",
    payload: { index }
  };
}

// reducers.js
const game = (state = getInitialState(), action) => {
  switch (action.type) {
    case "MAKE_MOVE": {
      const { index } = action.payload;
      return {
        ...state,
        board: {
          ...state.board,
          [index]: state.currentPlayer.marker,
        }
      };
    }

    default:
      return state;
  }
```

```
// Game.js
function mapStateToProps({ board }) {
  return board;
}

function mapDispatchToProps(dispatch) {
  return {
    makeMove: (index) => {
      dispatch(makeMove(index));
    }
  };
}

export default connect(
  mapStateToProps,
  mapDispatchToProps
)(Game);
```

```
// Game.js
function mapStateToProps({ board }) {
  return board;
}

function mapDispatchToProps(dispatch) {
  return {
    makeMove: (index) => {
      dispatch(makeMove(index));
    }
  };
}

export default connect(
  mapStateToProps,
  mapDispatchToProps
)(Game);
```

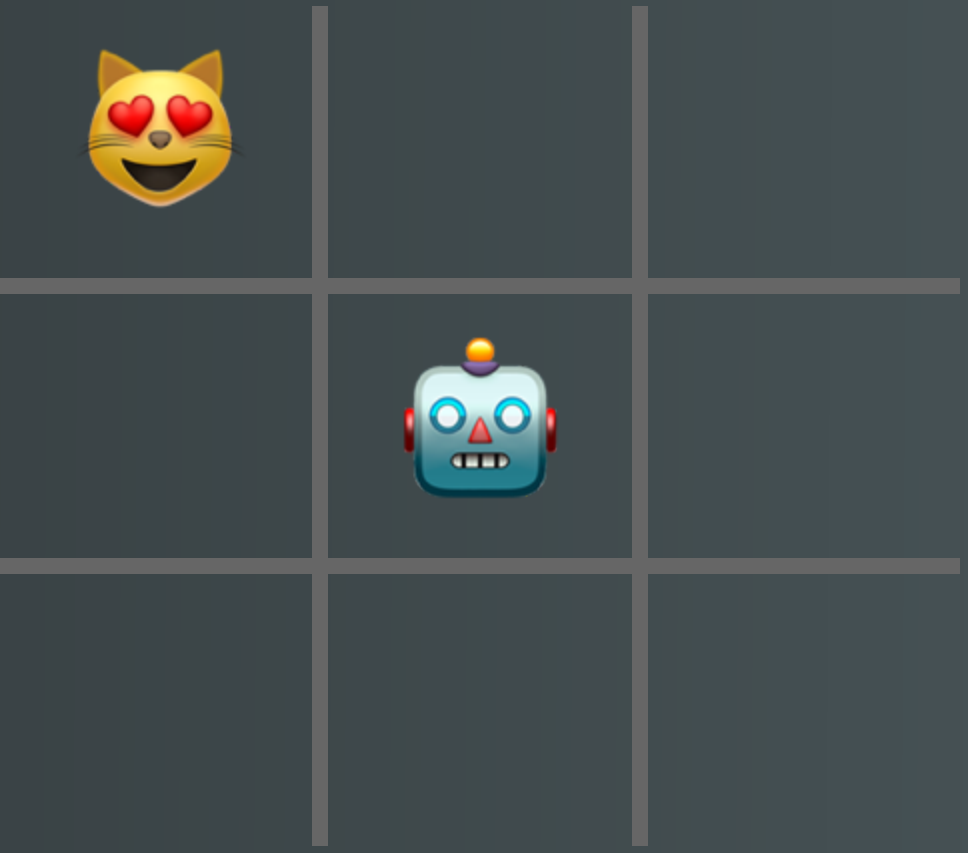
```
// Game.js
function mapStateToProps({ board }) {
  return board;
}

function mapDispatchToProps(dispatch) {
  return {
    makeMove: (index) => {
      dispatch(makeMove(index));
    }
  };
}

export default connect(
  mapStateToProps,
  mapDispatchToProps
)(Game);
```

@beccaliz



Becca played at spot 0

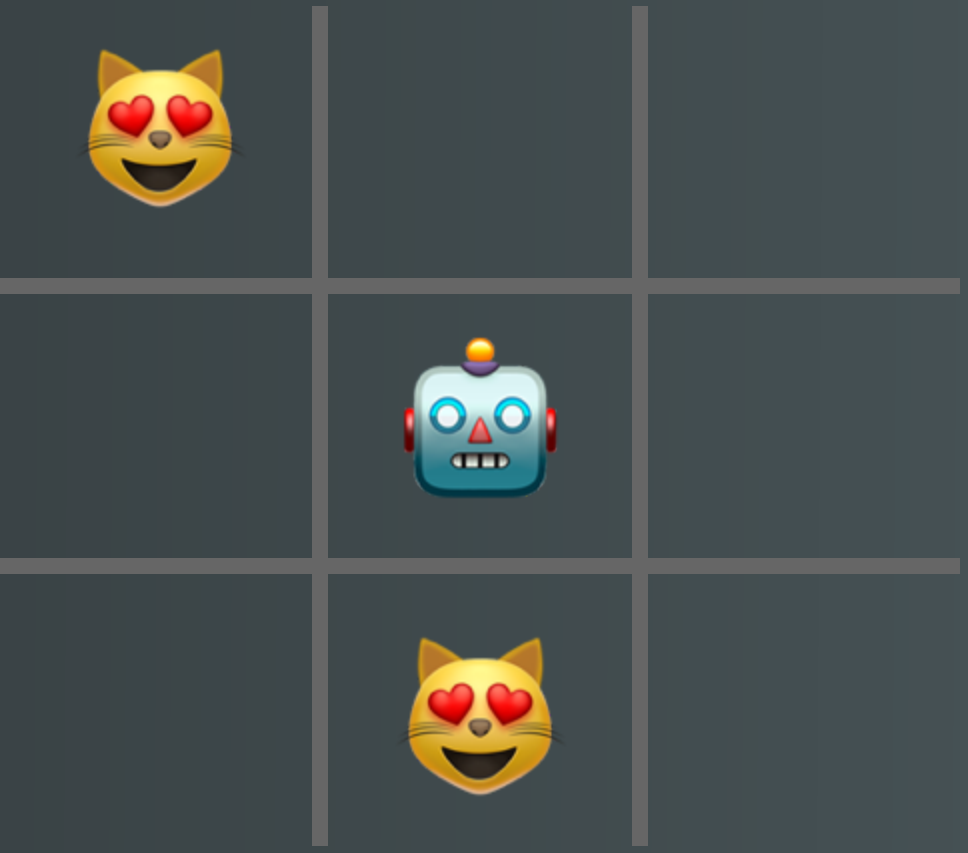
@beccaliz



Becca played at spot 0

Computer played at spot 4

@beccaliz



Becca played at spot 0

Computer played at spot 4

Becca played at spot 7

@beccaliz

Becca played at spot 0

Computer played at spot 4

Becca played at spot 7

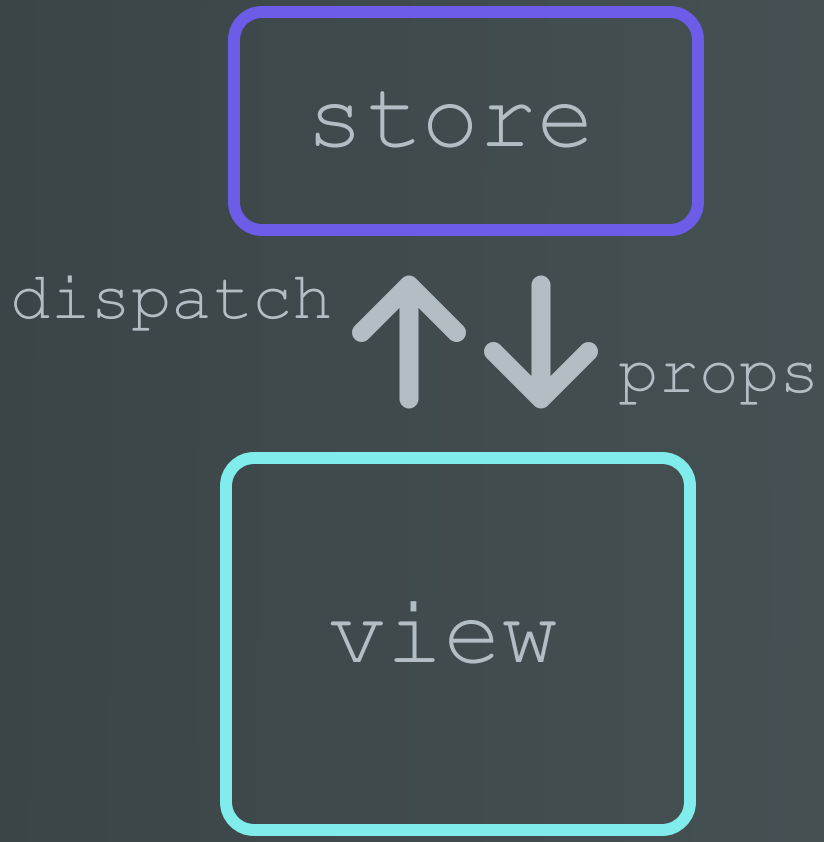
Computer played at spot 3

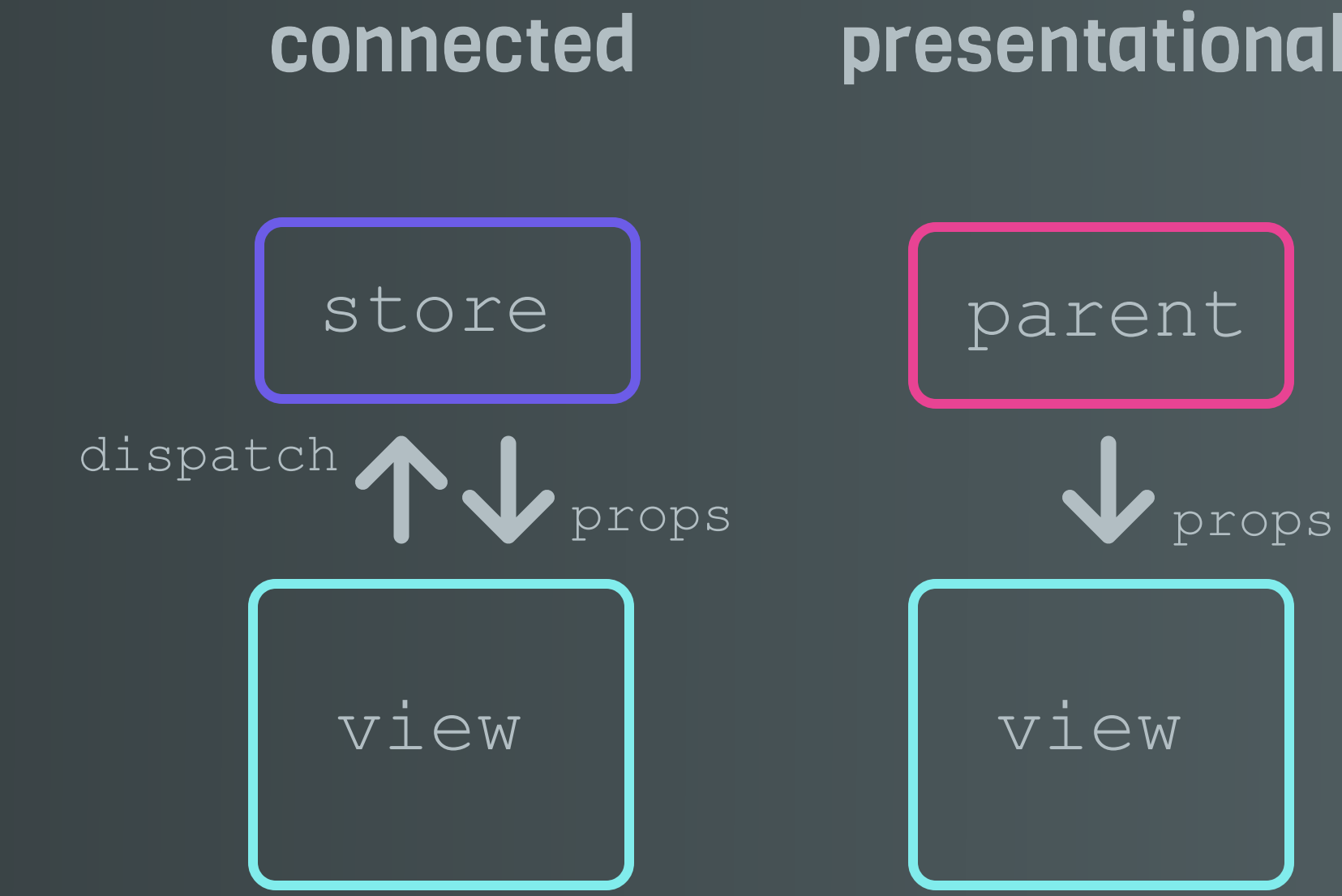
@beccaliz

```
// actions.js
export function makeMove(index) {
  return {
    type: "MAKE_MOVE",
    payload: { index }
  };
}

// reducers.js
const game = (state = getInitialState(), action) => {
  switch (action.type) {
    case "MAKE_MOVE": {
      const { index } = action.payload;
      return {
        ...state,
        board: {
          ...state.board,
          [index]: state.currentPlayer.marker,
        }
      };
    }

    default:
      return state;
  }
```

@beccaliz

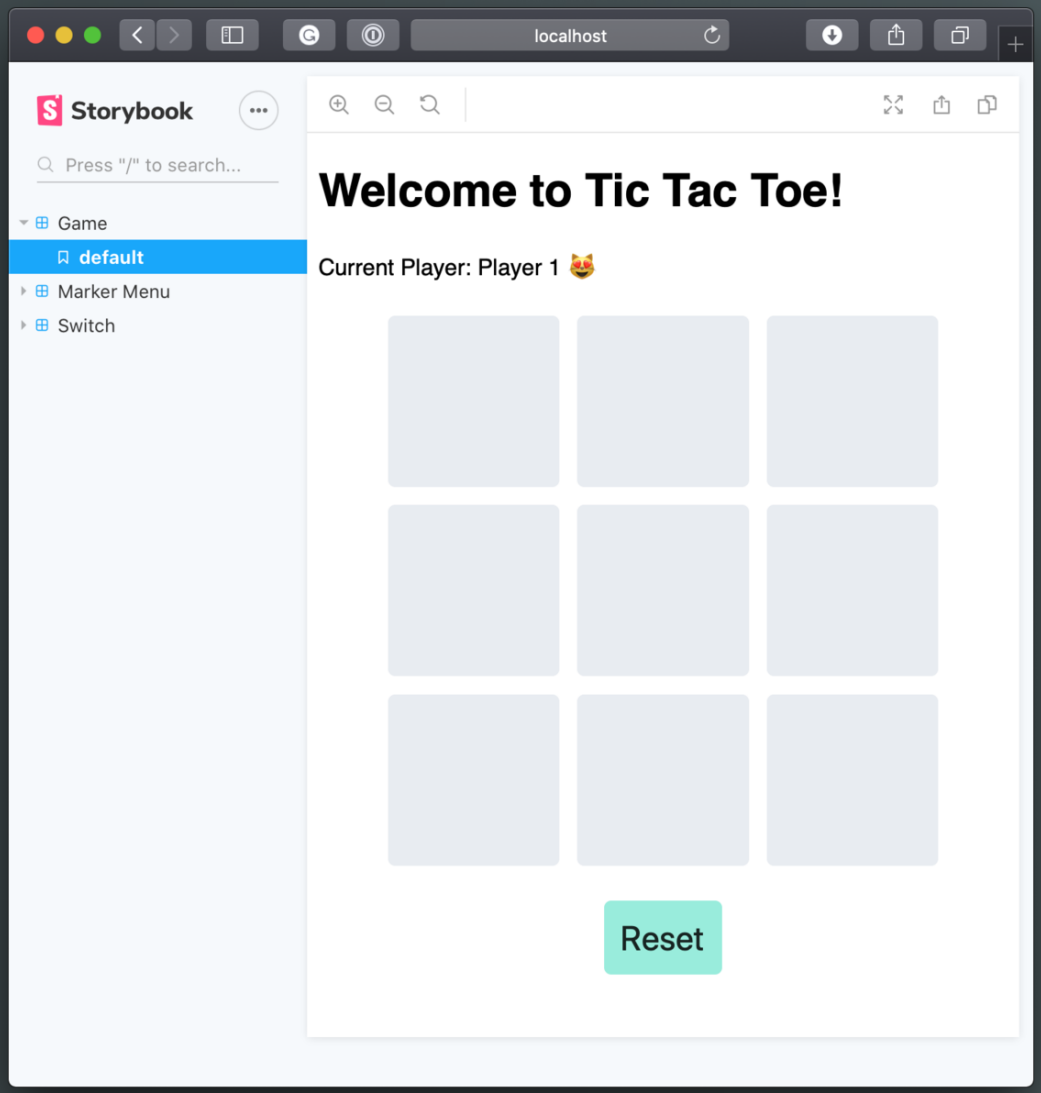
connected




connected
presentational
store
dispatch
props
view
parent
props
view

👍 Separation of Concerns 👍

👎 Too much abstraction 👎


localhost
Storybook
Press "/" to search...
Game
default
Marker Menu
Switch
Welcome to Tic Tac Toe!
Current Player: Player 1
Reset

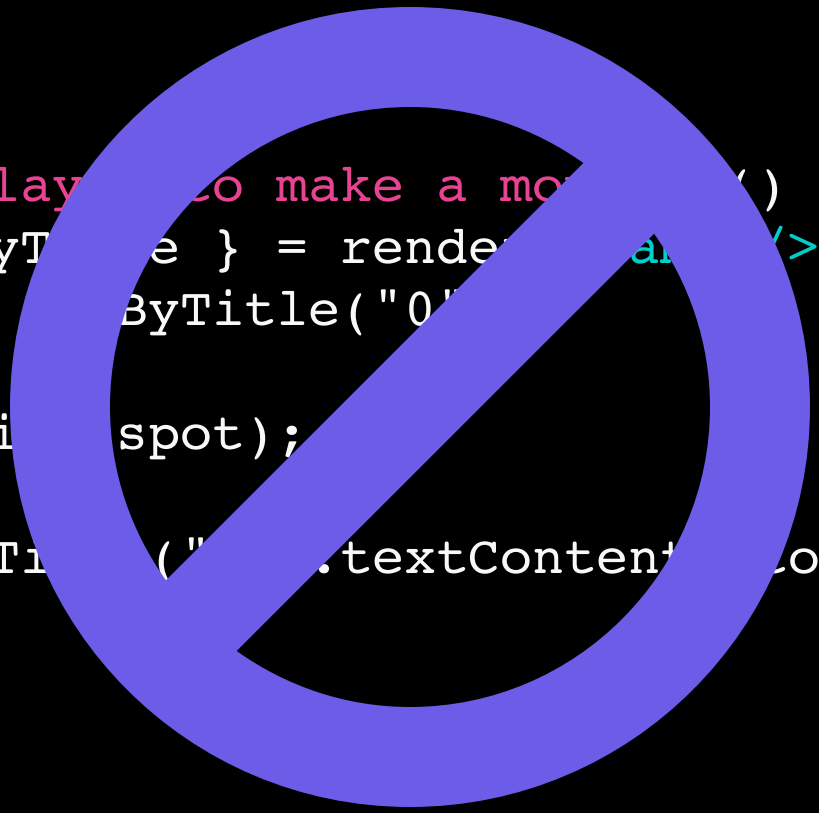
@beccaliz

```tsx
// Board.test.tsx

it("handles click events", () => {
  const props = {
    makeMove: jest.fn(),
    board: DEFAULT_BOARD
  };

  const { queryAllByRole } = render(<Board {...props}></Board>);

  fireEvent.click(queryAllByRole("button")[0]);

  expect(props.makeMove).toHaveBeenCalledWith(0);
});
```

```
// reducers.test.js

it("updates the board state", () => {
  expect(
    reducer(initialState, {
      type: "MAKE_MOVE",
      payload: { index: 2 }
    })
  ).toEqual({
    ...initialState,
    board: createBoard(`
      - - X
      - - -
      - - -
    `),
  });
});
```

```
1 it("allows a player to make a move", () => {
2   const { getByTitle } = render(<Game />);
3   const spot = getByTitle("0");
4
5   fireEvent.click(spot);
6
7   expect(getByTitle("0").textContent).toEqual("😻");
8 });
```

```
1 it("allows a player to make a move", () => {
2   const { getByTitle } = render(<Game />);
3   const spot = getByTitle("0");
4
5   fireEvent.click(spot);
6
7   expect(getByTitle("0").textContent).toEqual("😻");
8 });
```


it("allows a play to make a mo () => {
const { getBy e } = rende ar />);
const spot = ByTitle("0
fireEvent.cli spot);
expect(getByTi (" .textConten oEqual("😻");
});

matwrites.com

Home About me Contact Privacy Policy

matwrites.com

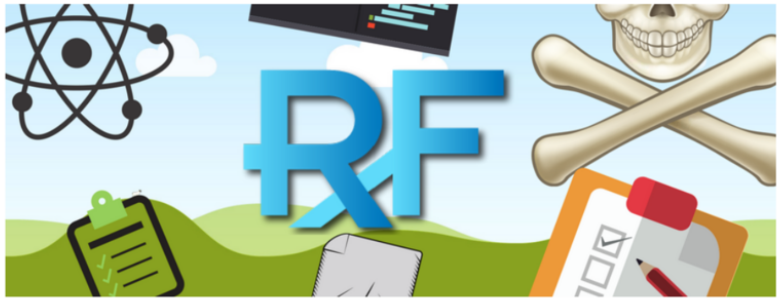
HOME REACT JAVASCRIPT REACT NATIVE NODE.JS ABOUT ME CONTACT

Home › React › Redux-form is dead

React

Redux-form is dead

By **Mateusz** - January 9, 2018 36667 13



Well, not exactly. But the project itself isn't doing very well and may kill your's app performance and act in a way which you don't expect it to. Here's quick and concise rant after rewriting the app to get rid of redux-form. And here's the reason why.

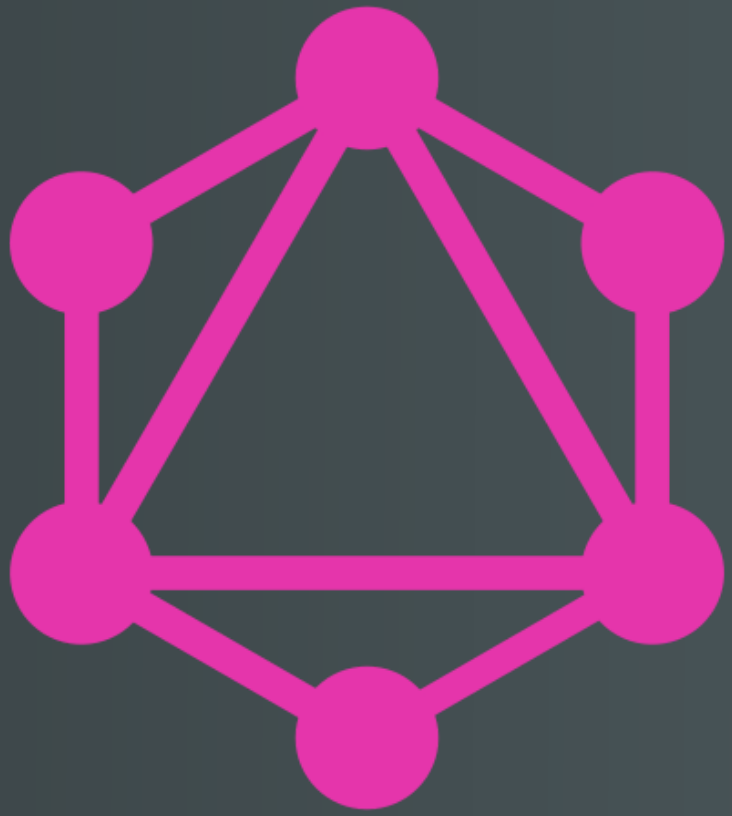
1,218 FOLLOW

349 FOLLOW

matwrites.com
285 likes
Like Page

Leave your e-mail address and join **500 other subscribers** who receive weekly, curated list of Frontend related links and news.

@beccaliz



@beccaliz

👍 Local State 👍

@beccaliz

👎 Prop Drilling 👎

👎 Prop Drilling 👎

👎 Duplication 👎

@beccaliz

✨ Higher Order Components ✨

@beccaliz

✨ Higher Order Components ✨

✨ Render Props ✨

```jsx
<ModalManager>
  {({ showModal, hideModal, visible }) => (
    <React.Fragment>
      <Button onClick={() => showModal()}>Click me!</Button>
      <Modal visible={visible}>
        <h1>You win!</h1>
        <Button onClick={() => hideModal()}>Close</Button>
      </Modal>
    </React.Fragment>
  )}
</ModalManager>
```

```jsx
<ModalManager>
  {({ showModal, hideModal, visible }) => (
    <React.Fragment>
      <Button onClick={() => showModal()}>Click me!</Button>
      <Modal visible={visible}>
        <h1>You win!</h1>
        <Button onClick={() => hideModal()}>Close</Button>
      </Modal>
    </React.Fragment>
  )}
</ModalManager>
```

```
const ModalManager = ({ children }) => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true)
  };

  function hideModal {
    setVisible(false)
  };

  render() {
    return (
      <React.Fragment>
        {children({ visible, showModal, hideModal })}
      </React.Fragment>
    )
  }
}
```

```
const ModalManager = ({ children }) => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true)
  };

  function hideModal {
    setVisible(false)
  };

  render() {
    return (
      <React.Fragment>
        {children({ visible, showModal, hideModal })}
      </React.Fragment>
    )
  }
}
```

```
<Query query={LOGGED_IN_USER}>
  {({ loading, error, data }) => {
    if (loading) {
      return <Spinner />;
    }
    if (error) {
      return <Error message={error} />;
    }
    return <Profile user={data}/>;
  }}
</Query>
```

```
<Query query={LOGGED_IN_USER}>
  {({ loading, error, data }) => {
    if (loading) {
      return <Spinner />;
    }
    if (error) {
      return <Error message={error} />;
    }
    return <Profile user={data}/>;
  }}
</Query>
```

```
<Query query={LOGGED_IN_USER}>
  {({ loading, error, data }) => {
    if (loading) {
      return <Spinner />;
    }
    if (error) {
      return <Error message={error} />;
    }
    return <Profile user={data}/>;
  }}
</Query>
```

```
<Query query={LOGGED_IN_USER}>
  {({ loading, error, data }) => {
    if (loading) {
      return <Spinner />;
    }
    if (error) {
      return <Error message={error} />;
    }
    return <Profile user={data}/>;
  }}
</Query>
```

```
<Container>
  <Query query={LOGGED_IN_USER}>
    {({ loading, error, data }) => {
      if (data) {
        return (
          <ModalManager>
            {({ showModal, hideModal, visible }) => {
              return (
                <React.Fragment>
                  <Button onClick={() => showModal()}>Click me!</Button>
                  {visible && (
                    <Modal>
                      <h1>Hello {data.user.name}!</h1>
                      <Button onClick={() => hideModal()}>Close</Button>
                    </Modal>
                  )}
                </React.Fragment>
              );
            }}
          </ModalManager>
        )
      }
    }}
  </Query>
</Container>
```

```
<Container>
  <Query query={LOGGED_IN_USER}>
    {({ loading, error, data }) => {
      if (data) {
        return (
          <ModalManager>
            {({ showModal, hideModal, visible }) => {
              return (
                <React.Fragment>
                  <Button onClick={() => showModal()}>Click me!</Button>
                  {visible && (
                    <Modal>
                      <h1>Hello {data.user.name}!</h1>
                      <Button onClick={() => hideModal()}>Close</Button>
                    </Modal>
                  )}
                </React.Fragment>
              );
            }}
          </ModalManager>
        )
      }
    }}
  </Query>
</Container>
```

```jsx
<Container>
  <Query query={LOGGED_IN_USER}>
    {({ loading, error, data }) => {
      if (data) {
        return (
          <ModalManager>
            {({ showModal, hideModal, visible }) => {
              return (
                <React.Fragment>
                  <Button onClick={() => showModal()}>Click me!</Button>
                  {visible && (
                    <Modal>
                      <h1>Hello {data.user.name}!</h1>
                      <Button onClick={() => hideModal()}>Close</Button>
                    </Modal>
                  )}
                </React.Fragment>
              );
            }}
          </ModalManager>
        )
      }
    }}
  </Query>
</Container>
```

```
<Container>
  <Query query={LOGGED_IN_USER}>
    {({ loading, error, data }) => {
      if (data) {
        return (
          <ModalManager>
            {({ showModal, hideModal, visible }) => {
              return (
                <React.Fragment>
                  <Button onClick={() => showModal()}>Click me!</Button>
                  {visible && (
                    <Modal>
                      <h1>Hello {data.user.name}!</h1>
                      <Button onClick={() => hideModal()}>Close</Button>
                    </Modal>
                  )}
                </React.Fragment>
              );
            }}
          </ModalManager>
        )
      }
    }}
  </Query>
</Container>
```

```
<Container>
  <Query query={LOGGED_IN_USER}>
    {({ loading, error, data }) => {
      if (data) {
        return (
          <ModalManager>
            {({ showModal, hideModal, visible }) => {
              return (
                <React.Fragment>
                  <Button onClick={() => showModal()}>Click me!</Button>
                  {visible && (
                    <Modal>
                      <h1>Hello {data.user.name}!</h1>
                      <Button onClick={() => hideModal()}>Close</Button>
                    </Modal>
                  )}
                </React.Fragment>
              );
            }}
          </ModalManager>
        )
      }
    }}
  </Query>
</Container>
```

```
// Game.js
function mapStateToProps({ board }) {
  return board;
}

function mapDispatchToProps(dispatch) {
  return {
    makeMove: (index) => {
      dispatch(makeMove(index));
    }
  };
}

export default connect(
  mapStateToProps,
  mapDispatchToProps
)(Game);
```

```
export default withRouter(
  withTheme(
    withSomeOtherState(
      connect(
        mapStateToProps,
        mapDispatchToProps
      )(Game)
    )
  )
);
```

```
export default withRouter(
  withTheme(
    withSomeOtherState(
      connect(
        mapStateToProps,
        mapDispatchToProps
      )(Game)
    )
  )
);
```

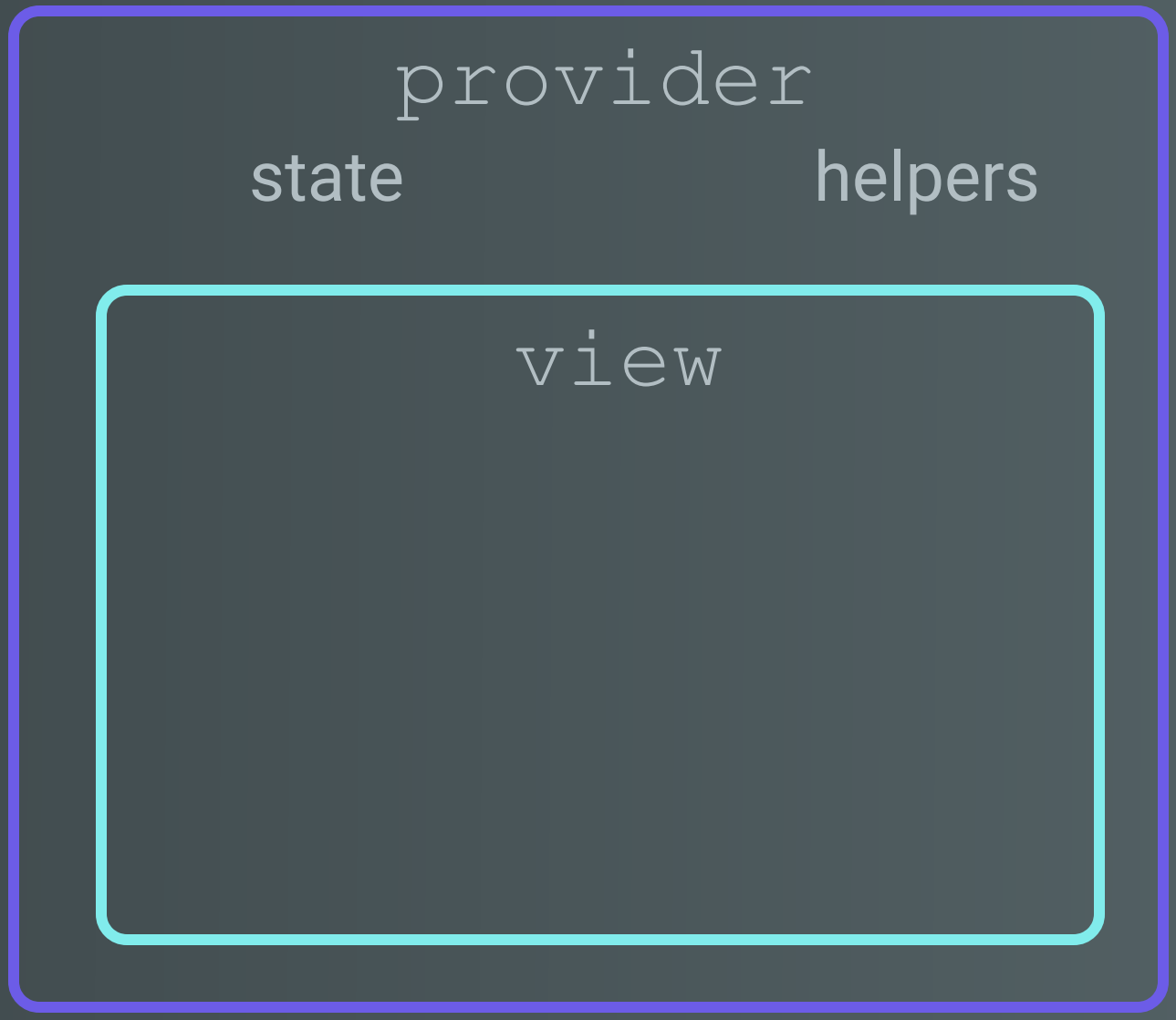
```
export default withRouter(
  withTheme(
    withSomeOtherState(
      connect(
        mapStateToProps,
        mapDispatchToProps
      )(Game)
    )
  )
);
```

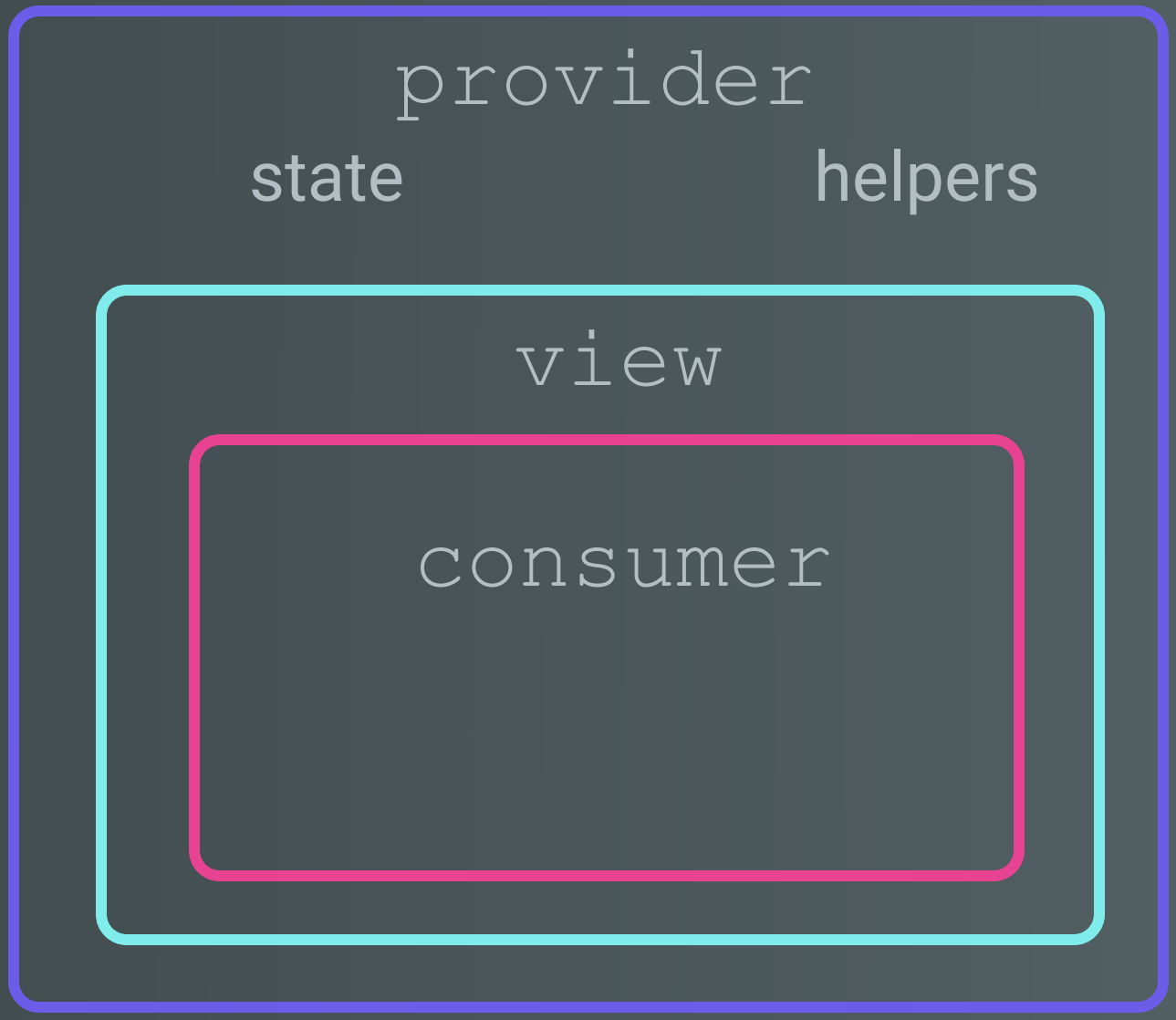
```
export default withRouter(
  withTheme(
    withSomeOtherState(
      connect(
        mapStateToProps,
        mapDispatchToProps
      )(Game)
    )
  )
);
```

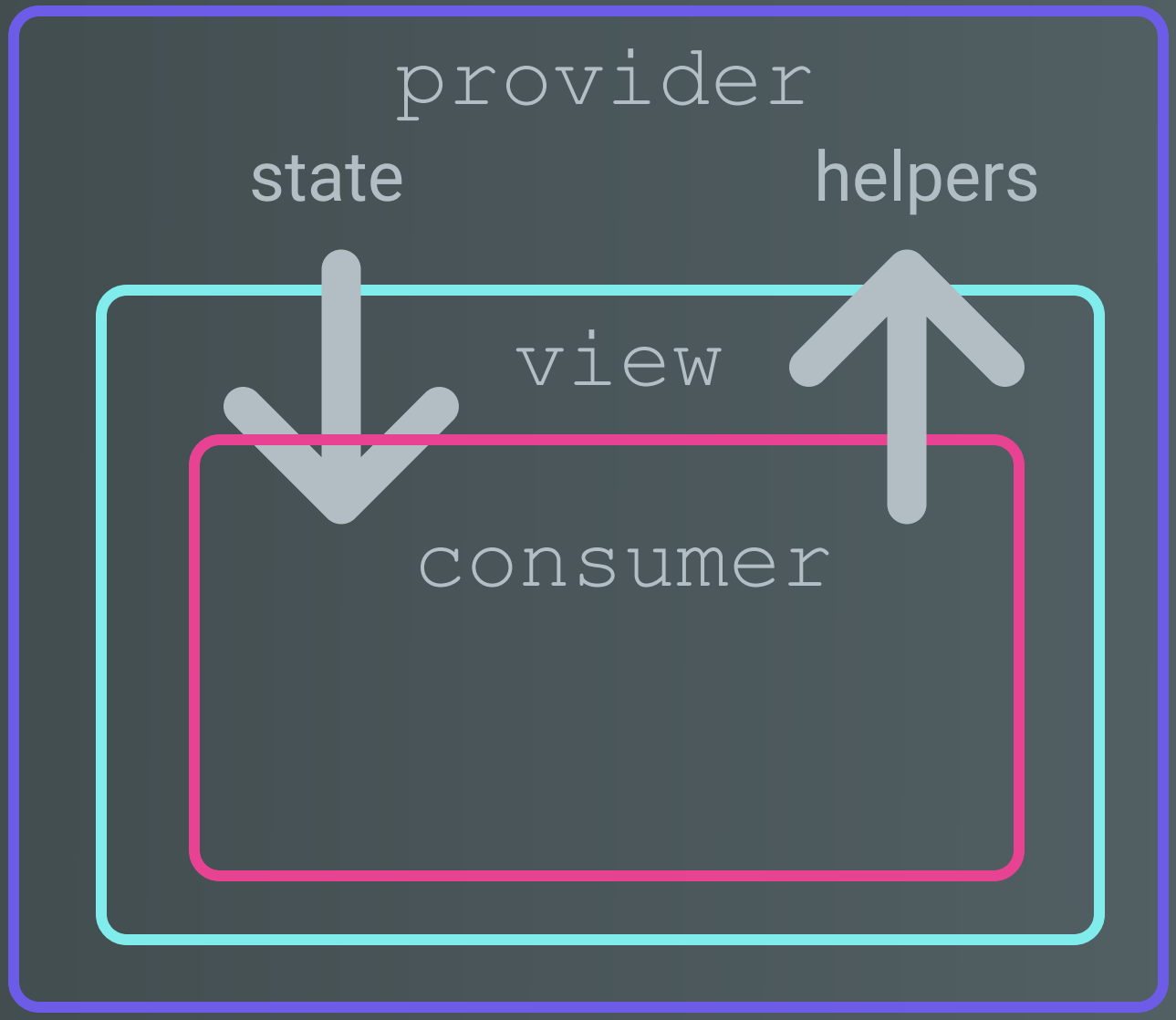
✨ Context ✨

@beccaliz

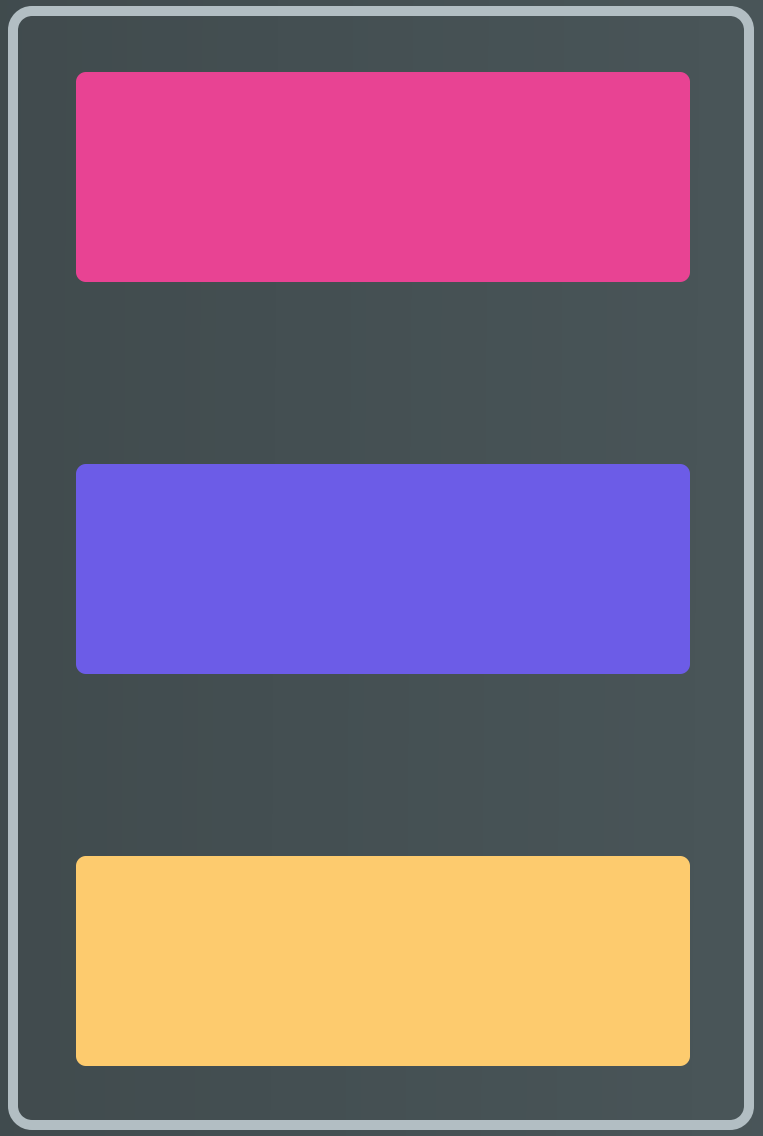

provider
state
helpers


provider
state
helpers
view

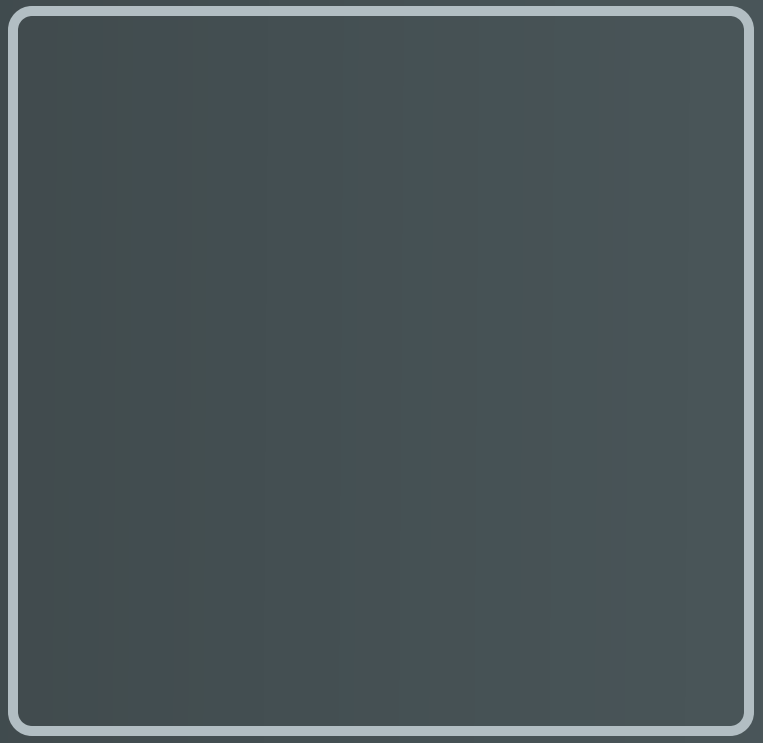

provider
state
helpers
view
consumer

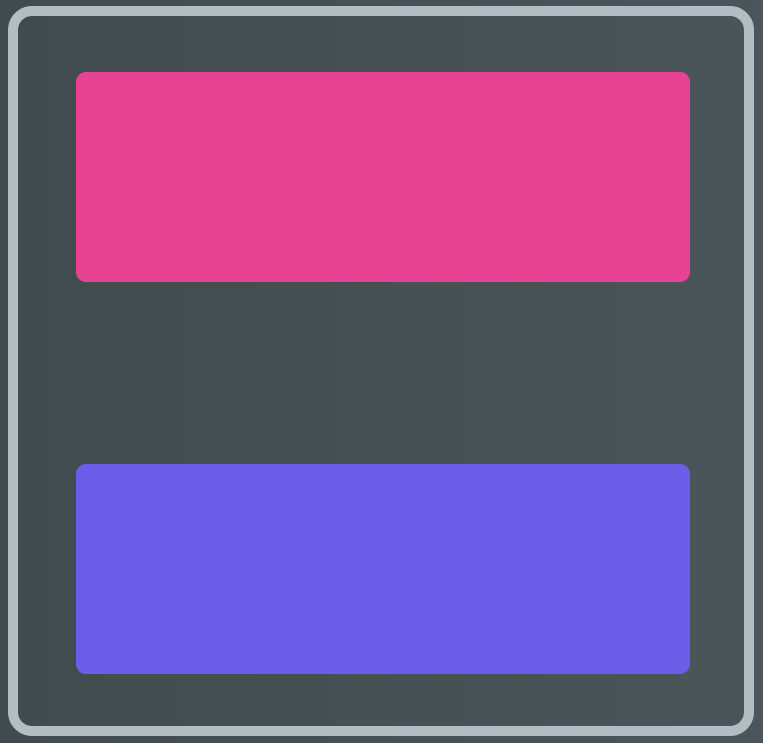

provider
state
helpers
view
consumer

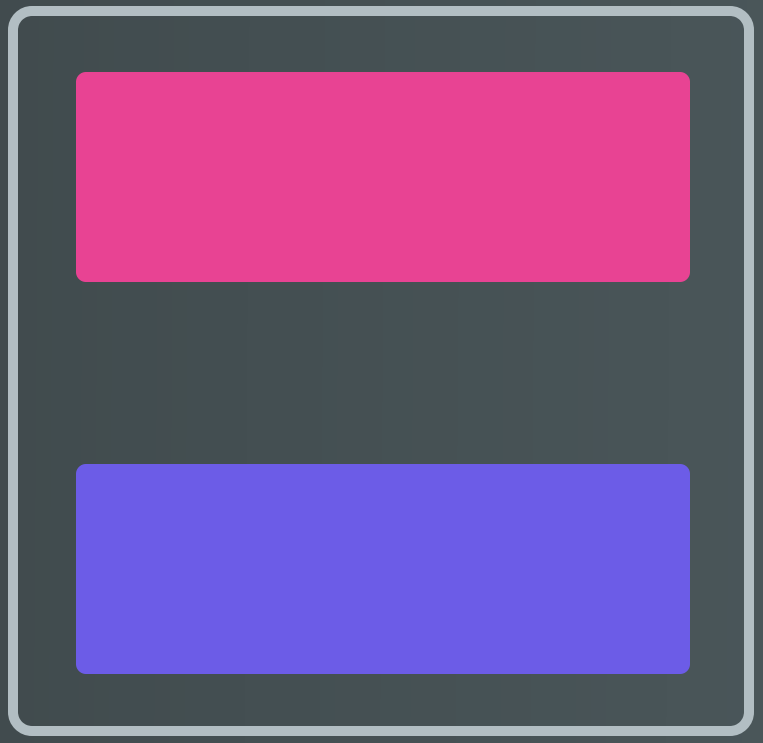
@beccaliz



context
provider







context
provider

```
1  // GreetingModal.js
2  function GreetingModal() {
3    const { user } = React.useContext(LoggedInUserContext);
4    const { hideModal } = React.useContext(ModalContext);
5
6    return (
7      <Modal id="greeting">
8        <h1>Hello {user.name}!</h1>
9        <Button onClick={() => hideModal()}>Close</Button>
10     </Modal>
11   )
12 }
```

```
1  // GreetingModal.js
2  function GreetingModal() {
3    const { user } = React.useContext(LoggedInUserContext);
4    const { hideModal } = React.useContext(ModalContext);
5
6    return (
7      <Modal id="greeting">
8        <h1>Hello {user.name}!</h1>
9        <Button onClick={() => hideModal()}>Close</Button>
10     </Modal>
11   )
12 }
```

```
const ModalProvider = ({ children }) => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true)
  };

  function hideModal {
    setVisible(false)
  };

  render() {
    return (
      <ModalContext.Provider values={{ visible, showModal, hideModal }}>
        {children}
      </ModalContext.Provider>
    );
  }
}
```

```
const ModalProvider = ({ children }) => {
  const [visible, setVisible] = React.useState(false);

  function showModal() {
    setVisible(true)
  };

  function hideModal {
    setVisible(false)
  };

  render() {
    return (
      <ModalContext.Provider values={{ visible, showModal, hideModal }}>
        {children}
      </ModalContext.Provider>
    );
  }
}
```

✨ Hooks ✨

```
// reducers/game.js
function useGame(initialState) {
  const [game, dispatch] = React.useReducer(gameReducer);

  function makeMove(index) {
    return dispatch({ type: "MAKE_MOVE", payload: index }
  }

  return { game, makeMove };
}
```

```
// reducers/game.js
function useGame(initialState) {
  const [game, dispatch] = React.useReducer(gameReducer);

  function makeMove(index) {
    return dispatch({ type: "MAKE_MOVE", payload: index }
  }

  return { game, makeMove };
}
```

```
// reducers/game.js
function useGame(initialState) {
  const [game, dispatch] = React.useReducer(gameReducer);

  function makeMove(index) {
    return dispatch({ type: "MAKE_MOVE", payload: index }
  }

  return { game, makeMove };
}
```

```
function gameReducer(state, action) {
  switch (action.type) {
    case "MAKE_MOVE": {
      const index = action.payload;
      const { currentPlayer, players } = state;
      const nextPlayer = switchPlayer(currentPlayer, players);
      return {
        ...state,
        board: {
          [index]: currentPlayer.marker,
          currentPlayer: nextPlayer,
          //...etc
        }
      };
    }
    default: {
      return state;
    }
  }
}
```

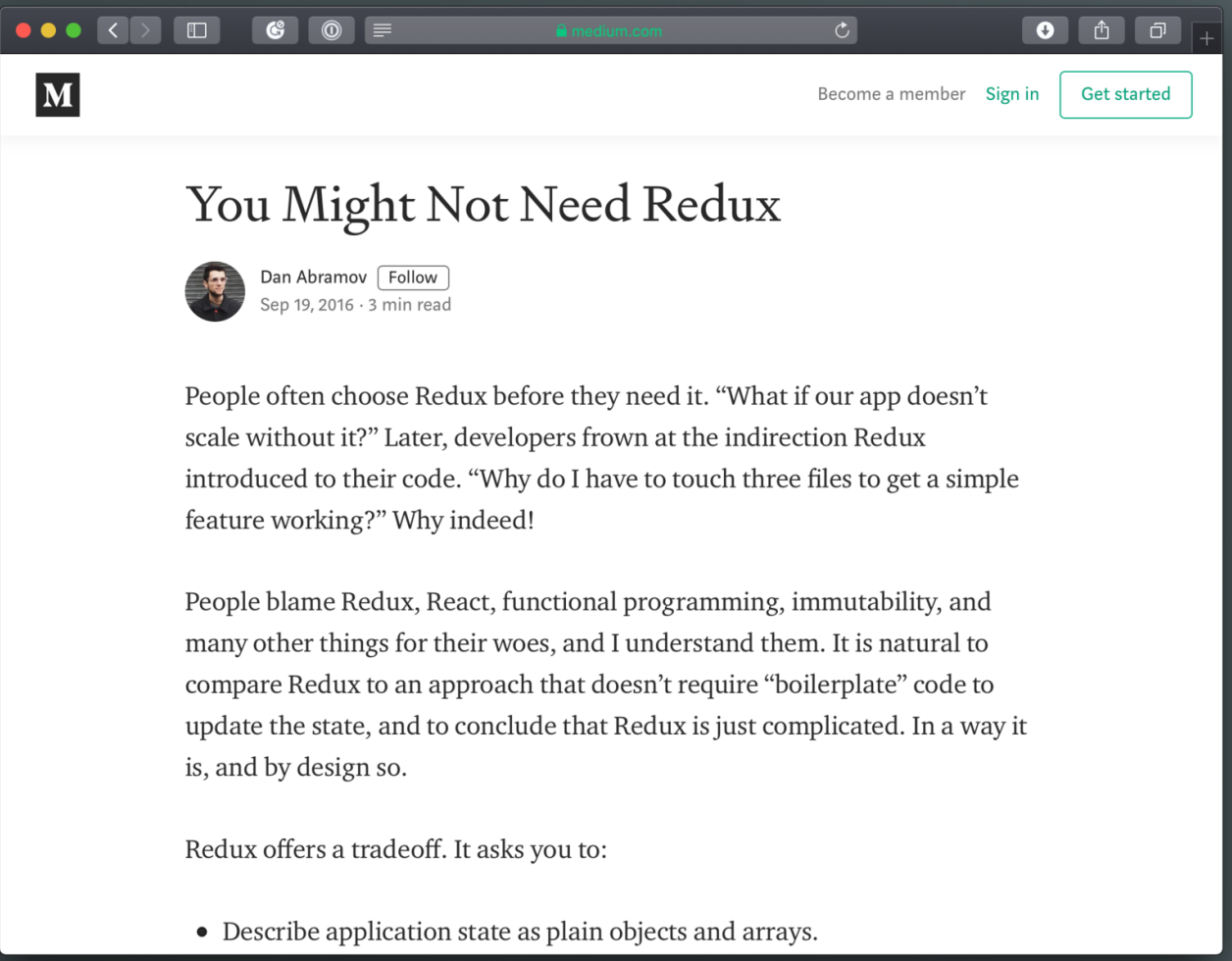
```
it("allows a player to make a move", () => {
  const { getByTitle } = render(<Game />);
  const spot = getByTitle("0");

  fireEvent.click(spot);

  expect(getByTitle("0").textContent).toEqual("😻");
});
```

Do you need Redux?

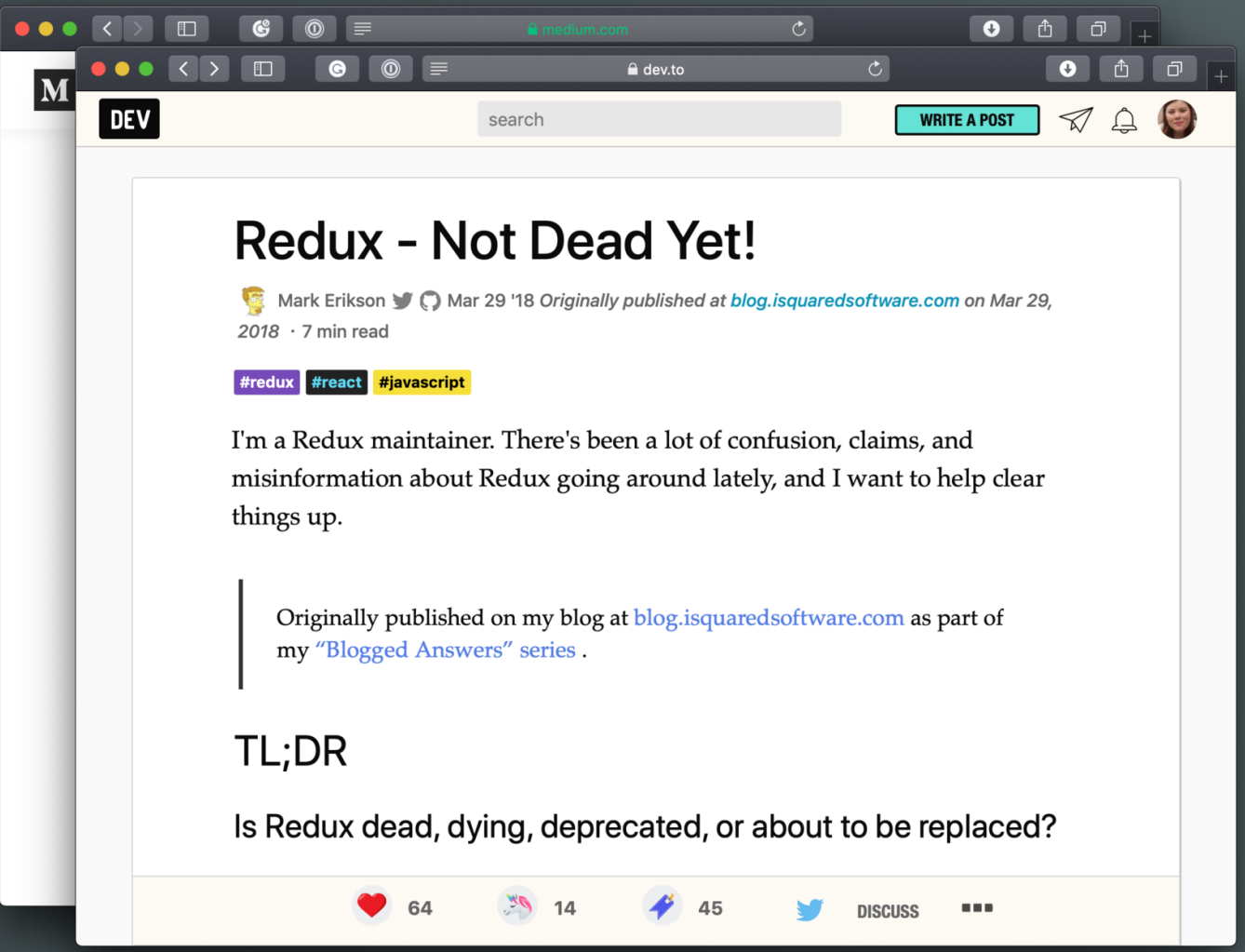
Complexity

MobX?

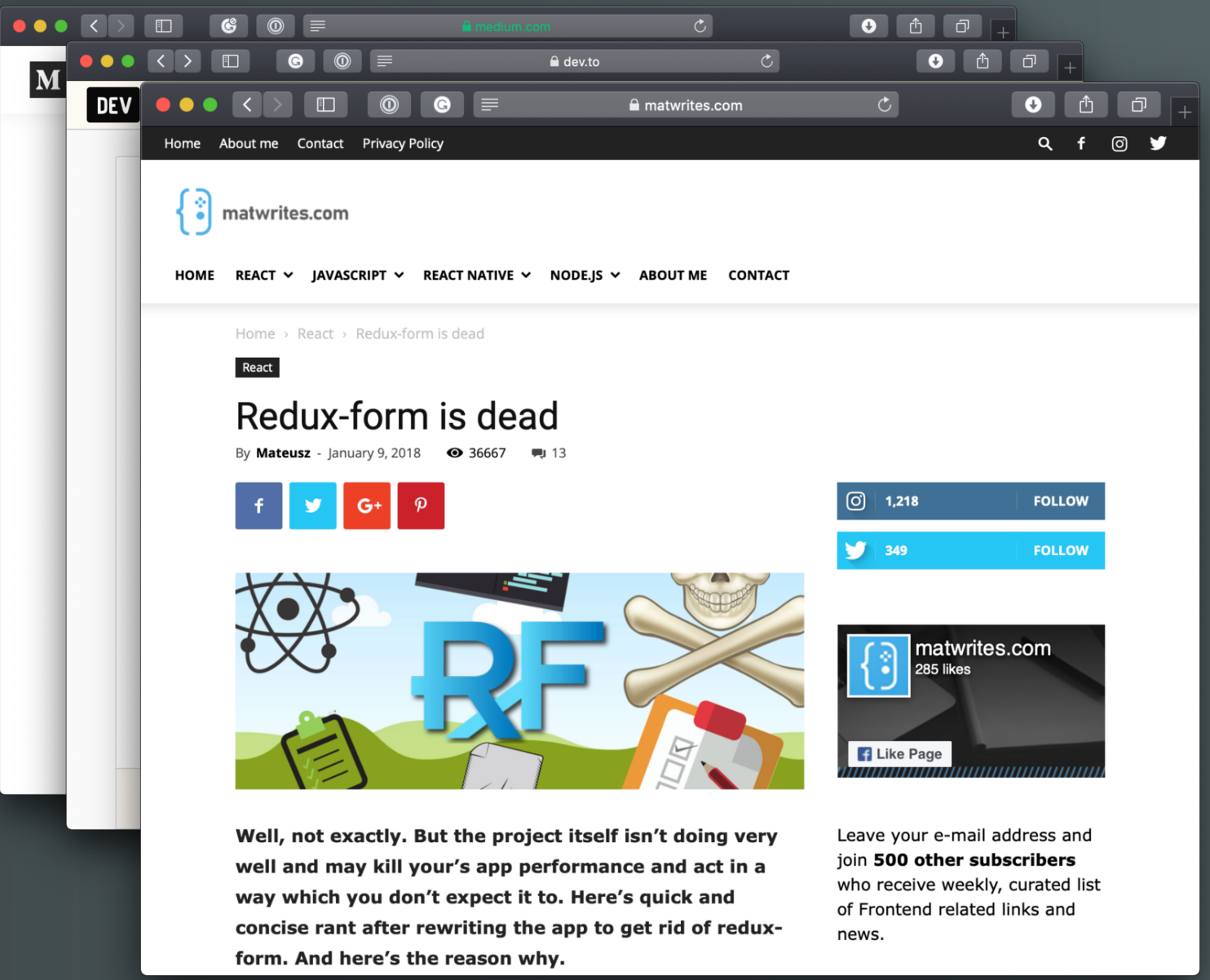
@beccaliz


medium.com
Become a member
Sign in
Get started
You Might Not Need Redux
Dan Abramov
Follow
Sep 19, 2016 · 3 min read
People often choose Redux before they need it. “What if our app doesn’t scale without it?” Later, developers frown at the indirection Redux introduced to their code. “Why do I have to touch three files to get a simple feature working?” Why indeed!
People blame Redux, React, functional programming, immutability, and many other things for their woes, and I understand them. It is natural to compare Redux to an approach that doesn’t require “boilerplate” code to update the state, and to conclude that Redux is just complicated. In a way it is, and by design so.
Redux offers a tradeoff. It asks you to:
Describe application state as plain objects and arrays.

@beccaliz


medium.com
dev.to
DEV
search
WRITE A POST
Redux - Not Dead Yet!
Mark Erikson Mar 29 '18 Originally published at blog.isquaredsoftware.com on Mar 29, 2018 · 7 min read
#redux #react #javascript
I'm a Redux maintainer. There's been a lot of confusion, claims, and misinformation about Redux going around lately, and I want to help clear things up.
Originally published on my blog at blog.isquaredsoftware.com as part of my "Blogged Answers" series .
TL;DR
Is Redux dead, dying, deprecated, or about to be replaced?
64
14
45
DISCUSS

@beccaliz


medium.com
dev.to
matwrites.com
Home
About me
Contact
Privacy Policy
matwrites.com
HOME
REACT
JAVASCRIPT
REACT NATIVE
NODE.JS
ABOUT ME
CONTACT
Home › React › Redux-form is dead
React
Redux-form is dead
By Mateusz - January 9, 2018
36667
13
RF
1,218
FOLLOW
349
FOLLOW
matwrites.com
285 likes
Like Page
Well, not exactly. But the project itself isn't doing very well and may kill your's app performance and act in a way which you don't expect it to. Here's quick and concise rant after rewriting the app to get rid of redux-form. And here's the reason why.
Leave your e-mail address and join 500 other subscribers who receive weekly, curated list of Frontend related links and news.

@beccaliz

How do we choose?

You don't have to choose just one.

What is the scope of the state?

Is there a commonly-used library that can help?

Am I repeating myself?

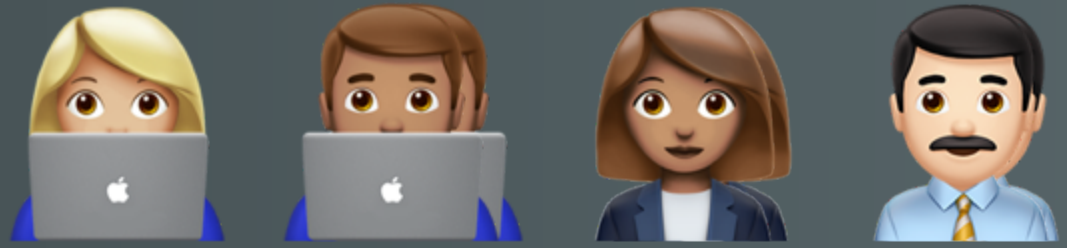
Incremental Changes

@beccaliz

@beccaliz

@beccaliz

@beccaliz

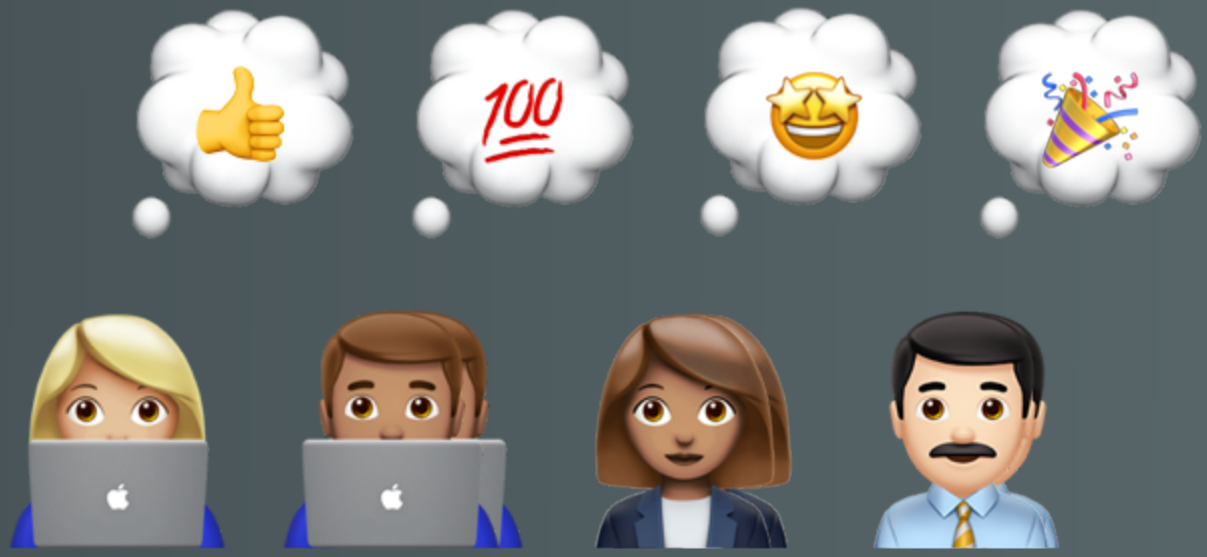


@beccaliz

@beccaliz

@beccaliz

@beccaliz



@beccaliz

The Newbie

The Loyalist

The Explorer

✨ Thank you!! ✨

 @beccaliz

 becca.is

Slides: