

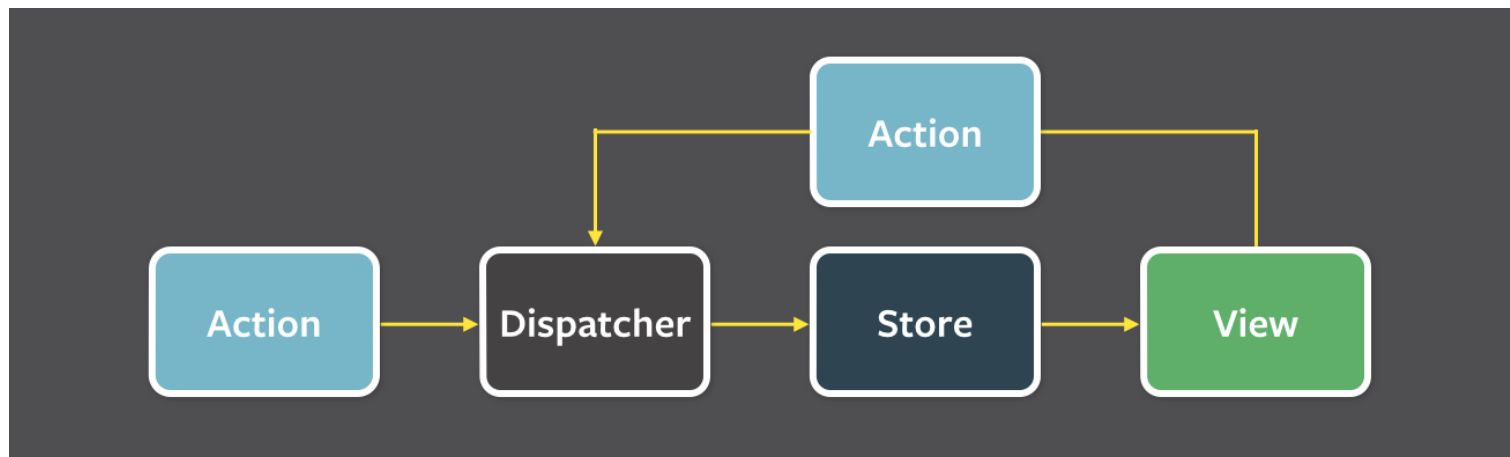
**choo**

**TDAH Talks S01E04 - São Carlos, Agosto 2016**

**Fabricio C Zuardi**

# Front-end hoje

- Single Page Applications
- Composable view components
- Unidirectional data flow
  - State -> View -> Action



# React

## the good parts

- JSX
- testable pieces
- help bringing functional programming to mainstream / hype
- redux

## the bad parts

- JSX, boilerplates, overhead
- People think Webpack is necessary
  - Webpack sucks hard

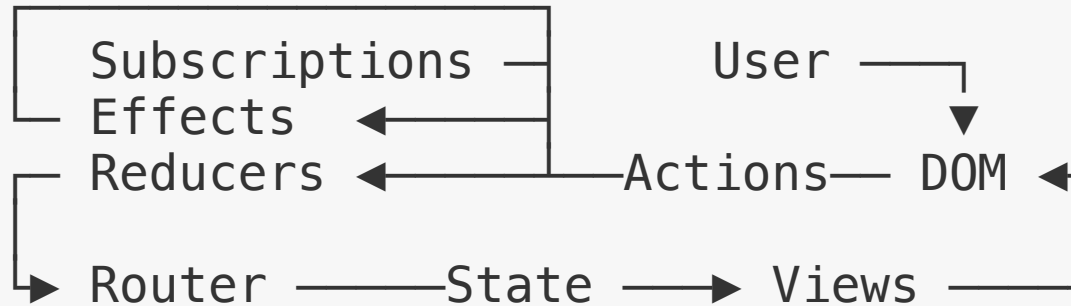
**Thanks React!**

**now let's take a step back**

# choo

- DOM the lowest common denominator
- Plain Javascript
  - Tagged template literals
- Useful concepts from flux/elm-architecture/redux/react-router organized in a small API (6 methods)
- transparent side-effects
- minimal tooling (browserify)

## choo (ascii-art)



é um tremzinho!

**Exemplo**

# Próximo TDAH

- data
- temas



# State

```
const exampleState = {  
  day: '09 de Setembro de 2016',  
  talks: [  
    {  
      title: 'Pokémon Go é Vida',  
      author: 'Frederico Marques'  
    },  
    {  
      title: 'Drinks a base de Mountain Dew',  
      author: 'Luiz Marquetti'  
    }  
  ]  
}
```

# Components

```
const dayLabel = day => html`<h2>dia ${day}</h2>`;
```

```
const talkItem = talk => html`  
  <li>  
    <span>${talk.title}</span> -  
    <span>${talk.author}</span>  
  </li>`;
```

```
const talkList = talks => html`  
  <ul>  
    ${talks.map(talkItem)}  
  </ul>`;
```

# View

```
const nextTDAH = state => html`  
  <div>  
    <h1>Próximo TDAH</h1>  
    ${dayLabel(state.day)}  
    <h3>Temas</h3>  
    ${talkList(state.talks)}  
  </div>`;
```

# Reducers

```
const addTalk = (state, data) => ({  
  ...state,  
  talks: [  
    ...state.talks,  
    {  
      title: data.title,  
      author: data.author  
    }  
  ]  
});
```

```
const removeTalk = (state, data) => ({  
  ...state,  
  talks: [  
    ...state.talks.slice(0, data.index),  
    ...state.talks.slice(data.index + 1)  
  ]  
});
```

```
const replaceTalks = (state, data) =>  
  ({...state, talks: data.talks });
```

# Model

```
const tdahModel = {  
  state: exampleState,  
  reducers: {  
    addTalk,  
    removeTalk,  
    replaceTalks  
  },  
  effects: {  
    saveList  
  },  
  subscriptions: {  
    loadList  
  }  
}
```

# App

```
const choo = require('choo');  
const app = choo();  
app.model(tdahModel);  
app.router([  
  ['/', nextTDAH]  
]);  
const tree = app.start();  
document.body.appendChild(tree)
```

# Obrigado

## Referências:

- <https://github.com/yoshuawuyts/choo>
- <https://facebook.github.io/react/index.html>
- <http://redux.js.org/>
- <http://guide.elm-lang.org/architecture/>

(tem mais 2 slides extras)

# Effects

```
const saveList = (data, state, send, done) => {  
  fetch('/talks', {  
    method: 'POST',  
    headers: {  
      'Accept': 'application/json',  
      'Content-Type': 'application/json'  
    },  
    body: JSON.stringify({talks: state.talks})  
  }).then(done);  
}
```



# Subscriptions

```
const loadList = (send, done) => {  
  fetch('/talks', {  
    method: 'GET',  
    headers: {  
      'Accept': 'application/json',  
      'Content-Type': 'application/json'  
    },  
    body: JSON.stringify({talks: state.talks})  
  }).then(data => {  
    send('replaceTalks', {talks: data.talks});  
    done();  
  });  
}
```