React

Integrantes:

- Francisco Olivares
- Gabriel Valenzuela

Router

```
import React from 'react'
import { BrowserRouter as Router, Route, Link } from "react-router-dom";
import Catch from "./views/Catch";
import Team from "./views/Team";
import Collection from "./views/Collection";
...
const App = () => {
  return (
    <Router>
      <div>
        <Header />
        <div className="box">
          <Route exact path="/" component={Catch} />
          <Route path="/team" component={Team} />
          <Route path="/collection" component={Collection} />
        </div>
      </div>
    </Router>
  )
}
```

Estructura general de componentes (DogTeam)

```
import React, { Component, Fragment } from 'react'
import { connect } from "react-redux"
import ActionCreators from "../store/action"
class DogTeam extends Component {
    constructor(props) {
        super(props)

        this.state = {
        }
    }
    ...
    render() {...}
}

const mapStateToProps = (state) => {
    return {
        team: state.DogTeamReducer['team']
    }
}

export default connect(mapStateToProps, ActionCreators)(DogTeam)
```

Estructura general de *actions*

```
export const EQUIP = 'EQUIP'
...
export const dogEquip = (payload) => {
    return {
        type: types.EQUIP,
        payload: payload
    }
}
...
const ActionCreators = {
    ...DogCardActions,
    ...DogCollectionActions,
    ...DogTeamActions
}

export default ActionCreators
```

Estructura general de *reducers*

```
import * as types from '../action/action-types'
export const intitialState = {
    team:{
        first:{
            img : "https:...",
            atk : ...
        },
        second:{
            img : "https:...",
            atk : ""
        }
    }
}
const DogTeamReducer = (state = intitialState, action) => {
    switch (action.type) {
        case types.EQUIP:
            ...

        default:
            return state
    }

}

export default DogTeamReducer
```

Estructura general de *root reducer*

```
import { combineReducers } from "redux";
import DogCardReducer from './dogCard'
import DogTeamReducer from './dogTeam'
import DogCollectionReducer from './dogCollection'

const rootReducer = combineReducers({
    DogCardReducer,
    DogTeamReducer,
    DogCollectionReducer
})

export default rootReducer
```

Estructura de Store

```
import { createStore, applyMiddleware, compose } from "redux"

// Se importan los reducers
import rootReducer from './reducers'

// MiddleWare imports
import logger from 'redux-logger'

const store = createStore(
    rootReducer,
    compose(
        applyMiddleware(
            logger
        )
    )
)

export default store
```

Componentes

Equipo Doggo (Reducer - Initial State)

```
export const intitialState = {
    team:{
        first:{
            img : "...",
            atk : ["Piedra", "Papel", "Tijera"][Math.floor(Math.random() * 3)]
        },
        second:{
            img : "...",
            atk : ""
        }
    }
}
```

Cosas más relevantes

- Se setean urls básicas para el perro inicial y el otro miembro del equipo.
- El tipo del perro se selecciona al azar.

Equipo Doggo (Reducer Actions)

```
switch (action.type) {
    case types.EQUIP:
        const {team} = state
        if(action.payload['type'] === 'first'){
            return { ...state, team:{first: action.payload['change_dog'],second:team.second}}
        }
        else if(action.payload['type'] === 'second'){
            console.log("entre al if")
            return { ...state, team:{second: action.payload['change_dog'],first:team.first}}
        }

    default:
        return state
}
```

Cosas más relevantes

- Acción *EQUIP* permite cambiar los miembros del equipo, entregandole un objeto doggo en el payload.

Equipo Doggo (Component)

```
render() {
        const { team } = this.props
        const {first} = team
        const {second} = team
        return (
            <div>
                <div>
                    <Columns>
                        <Columns.Column>
                            <div>
                                <img src={first.img}/>
                            </div>
                            <div>
                                <p>Ataque: {first.atk}</p>
                            </div>
                        </Columns.Column>
                        <Columns.Column>
                            <div>
                                <img src={second.img}/>
                            </div>
                            <div>
                                {second.atk != "" && (<p>Ataque:{second.atk}</p>)}
                            </div>
                        </Columns.Column>
                    </Columns>
                </div>
                <div>
                    <br/>
                </div>
            </div>
        )
    }
```

Explorar Doggos (Reducer)

```
const intitialState = {
    current_dog: {
        img : "...",
        atk : ""
    }
}
...
switch (action.type) {
    case types.RANDOM:
        let atks = ["Piedra", "Papel", "Tijera"]
        let api_image = action.payload['img']
        let random_atk = atks[Math.floor(Math.random() * atks.length)]
        return { ...state, current_dog: {atk:random_atk, "img":api_image }}
    default:
        return state
}
```

Explorar Doggos (Component)

```
constructor(props) {
    super(props)

    this.state = {
        win:""
    }

    this.random = this.random.bind(this)
    this.explore = this.explore.bind(this)
    this.battle = this.battle.bind(this)
}
...
random(img) {
    this.props.randomDog({img:img})
}

async explore() {
    const response = await fetch('https://dog.ceo/api/breeds/image/random');
    const json = await response.json();
    this.random(json.message)
}
```

Explorar Doggos (Component)

```
battle(){
    const {first} = this.props.team
    const {second} = this.props.team
    const {current_dog} = this.props
    let win = false
    switch(current_dog.atk){
        case "Piedra":
            if (first.atk == "Papel" || second.atk == "Papel") {
                win=true
            }
        case "Papel":
            if (first.atk == "Tijera" || second.atk == "Tijera") {
                win=true
            }
        case "Tijera":
            if (first.atk == "Piedra" || second.atk == "Piedra") {
                win=true
            }
    }
    if (win) {
        this.setState({ win: "Has Capturado a este doggo!" })
        this.props.catchDog({current_dog})
    }
    else{
        this.setState({ win: "Has Perdido :c" })
    }
    this.explore()
}
```

Explorar Doggos (Component)

```
render() {
    const { current_dog } = this.props
    return (
        <div>
            <div>
                <h1>Busca al doggo que desees atrapar</h1>
            </div>
            <div>
                <Button color="primary" onClick={this.explore}>Explorar doggos</Button>
            </div>
            <div>
                <img src={current_dog.img} alt=""/>
            </div>
            <div>
                <Button color="danger" onClick={this.battle}>A Batallar!</Button>
            </div>
            <div>
                {this.state.win}
            </div>
            <div>
                <p className="indent"><br/></p>
            </div>



        </div>
    )
}
```

Explorar Doggos (Component)

```
const mapStateToProps = (state) => {
    return {
        current_dog: state.DogCardReducer['current_dog'],
        team: state.DogTeamReducer['team']
    }
}
```

Cosas Relevantes:

- Se encontró una forma de comunicar los states de los reducers.

Colección de Doggos (Reducer)

```
const intitialState = {
    collection:[
        {
            img:"...",
            atk: dogTeam.intitialState.team.first.atk
        }
    ]
}
switch (action.type) {
    case types.CATCH:
        let {collection} = state
        return {...state, collection: [...collection, action.payload['current_dog']]}
    default:
        return state
}
```

Colección de Doggos (Component)

```
constructor(props) {
    super(props)

    this.state = {
        assign:""
    }

}
...
change = (ev) => {
    let where = ""
    if(ev.target.id=="first"){
        where="primer"
    }
    else{
        where="segundo"
    }
    this.setState({assign:"Se ha añadido el doggo a tu equipo como "+where+" miembro"})
    const {collection} = this.props
    this.props.dogEquip({type: ev.target.id, change_dog: collection[ev.target.value]})
}
```

Colección de Doggos (Component)

```
render() {
    const { collection } = this.props
    return (
        <div>
            <Fragment>
                <Columns>
                    {collection.map(item => (
                        <Columns.Column>
                            <div>
                                <img src={item.img}/>
                            </div>
                            <div>
                                <p>{item.atk}</p>
                            </div>
                            <Columns>
                                <Columns.Column>
                                    <Button id="first" value={collection.indexOf(item)} onClick={this.change}>...</Button>
                                </Columns.Column>
                                <Columns.Column>
                                    <Button  id="second" value={collection.indexOf(item)} onClick={this.change}>...</Button>
                                </Columns.Column>
                            </Columns>

                        </Columns.Column>
                    ))}
                </Columns>
            </Fragment>
            <div>
                <p>{this.state.assign}</p>
            </div>
        </div>
    )
}
```