import React from "https://cdn.skypack.dev/react"

import ReactDOM from "https://cdn.skypack.dev/react-dom"

import { Provider, connect } from "https://cdn.skypack.dev/react-redux"

import { createStore, combineReducers, applyMiddleware } from "https://cdn.skypack.dev/redux"

const Q = 'Q', W = 'W', E = 'E', A = 'A', S = 'S', D = 'D', Z = 'Z', X = 'X', C = 'C';

const keyPress = 'keyPress', keyClick = 'keyClick';

const buttons = [

{

keyCode: 81,

keyTrigger: Q,

id: 'Heater-1',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Heater-1.mp3',

pos: 0,

pressed: false

},

{

keyCode: 87,

keyTrigger: W,

id: 'Heater-2',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Heater-2.mp3',

pos: 1,

pressed: false

},

{

keyCode: 69,

keyTrigger: E,

id: 'Heater-3',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Heater-3.mp3',

pos: 2,

pressed: false

},

{

keyCode: 65,

keyTrigger: A,

id: 'Heater-4',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Heater-4\_1.mp3',

pos: 3,

pressed: false

},

{

keyCode: 83,

keyTrigger: S,

id: 'Clap',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Heater-6.mp3',

pos: 4,

pressed: false

},

{

keyCode: 68,

keyTrigger: D,

id: 'Open-HH',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Dsc\_Oh.mp3',

pos: 5,

pressed: false

},

{

keyCode: 90,

keyTrigger: Z,

id: "Kick-n'-Hat",

url: 'https://s3.amazonaws.com/freecodecamp/drums/Kick\_n\_Hat.mp3',

pos: 6,

pressed: false

},

{

keyCode: 88,

keyTrigger: X,

id: 'Kick',

url: 'https://s3.amazonaws.com/freecodecamp/drums/RP4\_KICK\_1.mp3',

pos: 7,

pressed: false

},

{

keyCode: 67,

keyTrigger: C,

id: 'Closed-HH',

url: 'https://s3.amazonaws.com/freecodecamp/drums/Cev\_H2.mp3',

pos: 8,

pressed: false

}

];

const keyPressed = (key) => {

return {

type: keyPress,

key: key

}

};

const keyClicked = (key) => {

return {

type: keyClick,

key: key

}

};

function processArray(key) {

const customEvent = new CustomEvent('imprimir', { detail: { text: key.id }});

document.dispatchEvent(customEvent);

buttons[key.pos].pressed = !buttons[key.pos].pressed;

let sonido = new Audio(buttons[key.pos].url);

//buttons[key.pos].pressed ? sonido.play() : sonido.pause();

sonido.play();

return buttons;

}

const keysReducer = (state = buttons, action) => {

switch (action.type) {

case keyPress:

return processArray(action.key);

case keyClick:

return processArray(action.key);

default:

return state;

}

};

const store = createStore(keysReducer);

class Tecla extends React.Component {

constructor(props) {

super(props);

this.state = {

tecla: props.tecla,

}

document.addEventListener('keydown', (event) => {this.activateKey(event)});

}

activateKey(event) {

if (this.props.tecla.keyCode == event.keyCode)

this.props.submitKeyPressed(this.props.tecla);

}

clickTecla() {

this.props.submitKeyClicked(this.props.tecla);

}

render() {

return (

<div>

<button class="drum-pad" onClick={this.clickTecla.bind(this)}>{this.state.tecla.keyTrigger}</button>

</div>

);

}

}

class Display extends React.Component {

constructor(props) {

super(props);

this.state = {

text: 'Waiting...',

}

document.addEventListener('imprimir', (texto) => {this.imprimirPantalla(texto)});

}

imprimirPantalla(texto) {

this.state.text = texto.detail.text;

this.forceUpdate();

}

render() {

return (

<div>

<label id="display">{this.state.text}</label>

</div>

);

}

}

class Panel extends React.Component {

constructor(props) {

super(props);

}

render() {

return (

<div id="container">

<h1> It's a drum machine </h1>

<div id="drum-machine">

{this.props.botones.map( (tecla, idx) => {

return <Tecla submitKeyClicked = {this.props.submitKeyClicked} submitKeyPressed = {this.props.submitKeyPressed} tecla={tecla} onClick={() => {this.togglePlay()}}/>;})

}

</div>

<Display />

</div>

);

}

};

const mapStateToProps = (state) => {

return {botones: state}

};

const mapDispatchToProps = (dispatch) => {

return {

submitKeyPressed: (key) => {

dispatch(keyPressed(key))

},

submitKeyClicked: (key) => {

dispatch(keyClicked(key))

}

}

};

let Container = connect(mapStateToProps, mapDispatchToProps)(Panel);

class AppWrapper extends React.Component {

render() {

return (

<Provider store={store}>

<Container/>

</Provider>

);

}

};

ReactDOM.render(<AppWrapper/>, document.getElementById('root'));