# Redux Exercice

## Play around with Redux

* 1. Installation

Install Redux

1.2 Create store, action and dispatch the action

import { createStore } from 'redux'

let separateStore = ()=>{

console.log('---------UpdatedStore---------')

console.log(store.getState())

}

let store = createStore(

(store, action) => {

if (action.type === 'ADD\_ELEMENT') {

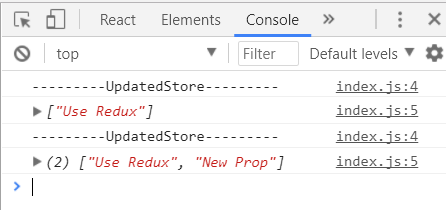
return store.concat(action.payload)

}

return store

},

['Use Redux']

)

separateStore()

store.dispatch({

type: 'ADD\_ELEMENT',

payload: ['New Prop']

})

separateStore()

* 1. Create counter

Create store and reducer that will be able to increment, decrement and clear the counter

import { createStore } from 'redux'

let separateStore = () => {

console.log(store.getState())

console.log('---------UpdatedStore---------')

}

let reducer = (store, action) => {

switch (action.type) {

case 'INCREMENT':

console.log('increment')

return ++store

case 'DECREMENT':

console.log('decrement')

return --store

case 'CLEAR':

console.log('clear')

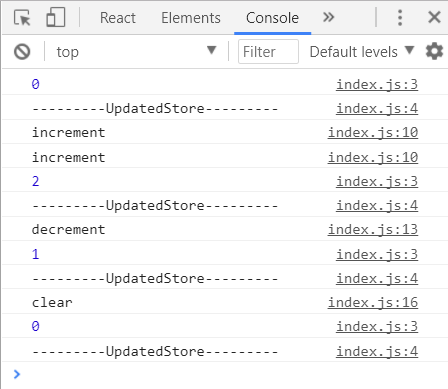
return 0

default:

return store

}

}

let store = createStore(reducer, 0)

separateStore()

store.dispatch({

type: 'INCREMENT'

})

store.dispatch({

type: 'INCREMENT'

})

separateStore()

store.dispatch({

type: 'DECREMENT'

})

separateStore()

store.dispatch({

type:'CLEAR'

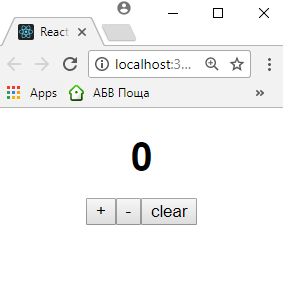
})

separateStore()

## Play around with redux and react

2.1 Create simple element

Create React Element, like the one from the picture below. Implement on click events, that will dispatch actions responding to the buttons from the example.



HINT: The following piece of code will take care of the rerendering for now.

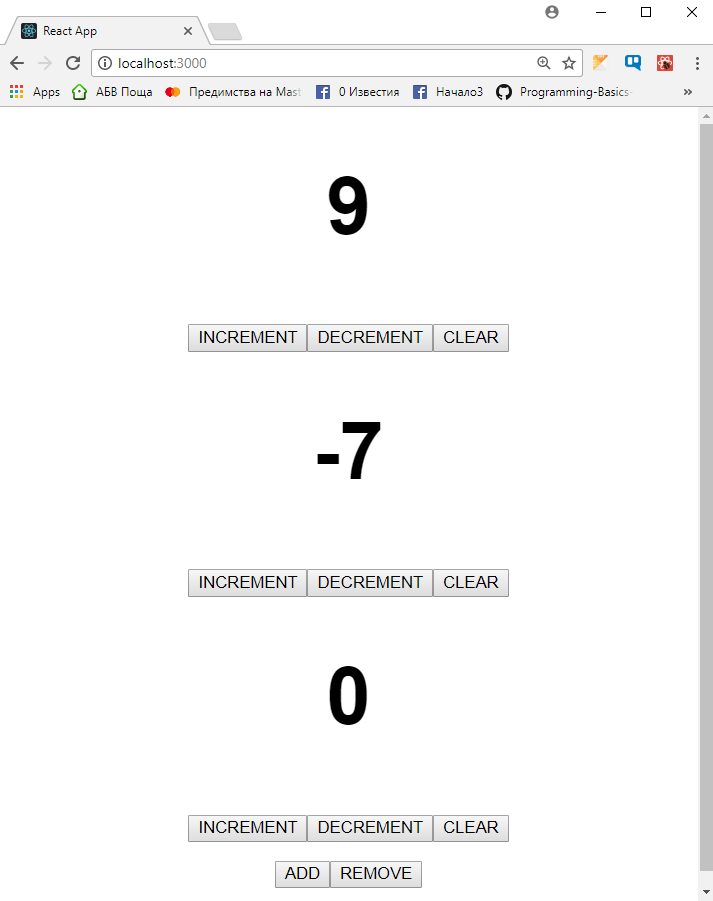
store.subscribe(()=>{

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

})

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

1.2 Implement logic that will allow 'addCounter' functionality

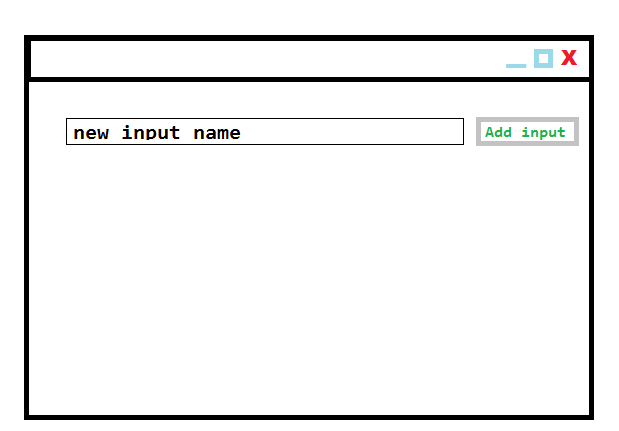


## Follow the User Story

Most of the time at work you will be given a user story and set of images, that must be turned in reality. Try to recreate the provided user story, using React and Redux

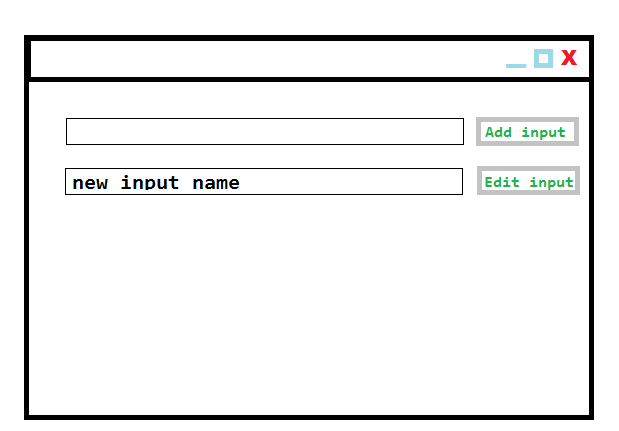
#### 3.1 Task one

When I land the home page I’m seeing the following view



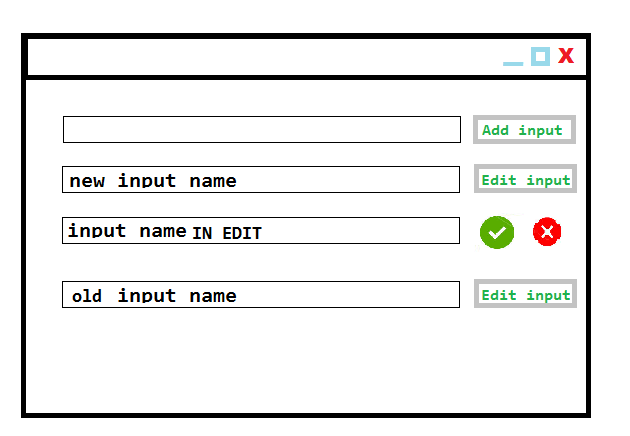
#### 3.2 Task two

I am able to add new inputs with the 'Add input' button



#### 3.3 Task three

I’m able to edit all inputs and while editing I have the option to discard or save the new 'Input' value



#### 3.4 Task four

I’m able to delete the last created 'Input'

