

main.jsx 공통



```
1  import { StrictMode } from 'react'
2  import { createRoot } from 'react-dom/client'
3  import './index.css'
4  import App from './App.jsx'
5
6  createRoot(document.getElementById('root')).render(
7    <StrictMode>
8      <App />
9    </StrictMode>,
10  )
```

App.jsx



```
1  import './App.css'
2  import { useState } from 'react'
3
4  function App() {
5    const [name, setName] = useState('')
6    const [year, setYear] = useState('')
7    const [warning, setWarning] = useState('')
8
9    const handleNameChange = (newName) => {
10      const formattedName = newName.trim().toLowerCase()
11      setName(formattedName)
12    }
13
14    const handleYearChange = (newYear) => {
15      const age = new Date().getFullYear() - newYear
16      if (age < 18) {
17        setWarning('Must be at least 18 yrs old!')
18      } else {
19        setWarning('')
20        setYear(newYear)
21      }
22    }
23
24    return (
25      <div>
26        <input
27          type="text"
28          placeholder="Enter name"
29          value={name}
30          onChange={
31            (e) => handleNameChange(e.target.value)
32          }
33        />
34        <input
35          type="number"
36          placeholder="Enter birth year"
37          value={year}
38          onChange={
39            (e) => handleYearChange(e.target.value)
40          }
41        />
42        {
43          warning && (
44            <p style={{ color: 'red' }}>{warning}</p>
45          )
46        }
47      )
48    )
49  }
```

```

45     <p>Name: {name}</p>
46     <p>Year: {year}</p>
47   </div>
48 )
49 }
50 export default App

```

결과

Must be at least 18 yrs old!

Name: test

현재 연도 입력

 Year: 202

1. useReducer

- 여러 컴포넌트에서 상태를 공유할 때 사용
- reducer는 "상태"와 "액션"을 받아서 "새로운 상태"를 반환하는 함수이다

App.jsx



```

1  import './App.css'
2  import { useState } from 'react'
3
4  // 1) useReducer
5  import React, { useReducer } from 'react' // {useReducer} 리액트에서 제공되는 기능
6  import { userReducer, initialState } from './reducers/userReducer'
7
8  function App() {
9    /*
10     * 1) useReducer - 여러 컴포넌트에서 상태를 공유할 때 사용
11     *   reducer는 "상태"와 "액션"을 받아서 "새로운 상태"를 반환하는 함수이다
12     */
13     const [state, dispatch] // 대부분 state, dispatch 라고 명시 (변수 이름은 변경가능)
14     = useReducer(userReducer, initialState) // useReducer(원수, 초기값)
15
16     return (
17       <div>
18         <input
19           type="text" placeholder="Enter name"
20           value={state.name}
21           onChange={(e) => dispatch({
22             // userReducer.js 에서
23             // type은 userReducer(state, action)에서 action값으로 입력됨
24             // payload는 userReducer(state, action)에서 return문의 action.payload로 입력됨
25             type: 'SET_NAME', payload: e.target.value })}
26         />
27         <input
28           type="number" placeholder="Enter birth year"
29           value={state.year}
30           onChange={(e) => dispatch({
31             type: 'SET_YEAR', payload: e.target.value })}
32         />
33         {state.warning
34           && <p style={{ color: 'red' }}>{state.warning}</p>
35         <p>Name: {state.name}</p>
36         <p>Year: {state.year}</p>

```

```

37     </div>
38   )
39 }
40 export default App

```

userReducer.js

JS

```

1  /*
2   * 1) useReducer - 여러 컴포넌트에서 상태를 공유할 때 사용
3   *   reducer는 "상태"와 "액션"을 받아서 "새로운 상태"를 반환하는 함수이다
4   */
5
6  // 반드시 reducers 폴더에 넣어야 되지 않지만, 보통 폴더를 만들어 관리한다
7  export const initialState = {
8    name: '',
9    year: '',
10   warning: ''
11 }
12
13 export function userReducer(state, action) {
14   switch (action.type) {
15     case 'SET_NAME':
16       return {
17         ...state,
18         name: action.payload.trim().toLowerCase()
19       }
20     case 'SET_YEAR':
21       const age
22       = new Date().getFullYear() - action.payload
23       if (age < 18) {
24         return {
25           ...state,
26           warning: 'Must be at least 18 yrs old!'
27         }
28       }
29       return {
30         ...state,
31         year: action.payload,
32         warning: ''
33       }
34     default:
35       throw new Error('Unknown action type')
36   }
37 }

```

결과

test

202

Must be at least 18 yrs old!

Name: test

Year: 202

현재 연도 입력

2. const [state,dispatch] = useReducer (userReducer,data,init) // data.js

App.jsx



```

1  import './App.css'

```

```

2  import { useState } from 'react'
3
4  // 2) data.js
5  import React, { useReducer } from 'react'
6  import { userReducer, init } from './reducers/userReducer'
7  import data from './data'
8
9  function App() {
10    // 2) data.js
11    const [state, dispatch]
12    // userReducer.js의 init은 init함수를 호출, data는 관리하던 값으로 들어감
13    = useReducer (userReducer, data, init)
14
15    return (
16      <div>
17        <input
18          type="text" placeholder="Enter name"
19          value={state.name}
20          onChange={(e) => dispatch({
21            // userReducer.js 에서
22            // type은 userReducer(state, action)에서 action값으로 입력됨
23            // payload는 userReducer(state, action)에서 return문의 action.payload로 입력됨
24            type: 'SET_NAME', payload: e.target.value })}
25        />
26        <input
27          type="number" placeholder="Enter birth year"
28          value={state.year}
29          onChange={(e) => dispatch({
30            type: 'SET_YEAR', payload: e.target.value })}
31        />
32        {state.warning
33          && <p style={{ color: 'red' }}>{state.warning}</p>}
34        <p>Name: {state.name}</p>
35        <p>Year: {state.year}</p>
36        { /* 2) data.js */}
37        <button onClick={
38          () => dispatch({ type: 'RESET', payload: data })}>
39          Reset
40        </button>
41
42      </div>
43    )
44  }
45  export default App

```

userReducer.js

JS

```

1  /*
2   * 1) useReducer - 여러 컴포넌트에서 상태를 공유할 때 사용
3   *   reducer는 "상태"와 "액션"을 받아서 "새로운 상태"를 반환하는 함수이다
4   */
5
6  // 반드시 reducers 폴더에 넣어야 되지 않지만, 보통 폴더를 만들어 관리한다
7  export const initialState = {
8    name: '',
9    year: '',
10    warning: ''
11  }
12
13  export function userReducer(state, action) {
14    switch (action.type) {
15      case 'SET_NAME':
16        return {

```

```

17     ... state,
18     name: action.payload.trim().toLowerCase() }
19   case 'SET_YEAR':
20     const age
21     = new Date().getFullYear() - action.payload
22     if (age < 18) {
23       return {
24         ... state,
25         warning: 'Must be at least 18 yrs old!' }
26     }
27     return {
28       ... state,
29       year: action.payload,
30       warning: '' }
31
32   // //2) data.jk
33   case 'RESET':
34     return init(action.payload)
35
36   default:
37     throw new Error('Unknown action type')
38   }
39 }
40
41 // 2) data.js
42 export function init(externalData) {
43   return {
44     ... initialState,
45     name: externalData.name,
46     year: externalData.year
47   }
48 }

```

data.js

JS

```

1 // 2) data.js
2 const externalData = {
3   name: 'jane doe',
4   year: 1995
5 }
6 export default externalData

```

결과

Name: jane doe

Year: 1995 초기값

Reset

3. countReducer.js

App.jsx



```

1 import './App.css'
2 import { useState } from 'react'

```

```

3
4 // 3) countReducer.js
5 import React, { useReducer } from 'react'
6 import { countReducer, initialState } from './reducers/countReducer'
7
8
9 function App() {
10   // 3) countReducer.js
11   const [state, dispatch]
12   = useReducer(countReducer, initialState)
13
14   const handleClick = (type, value, event) => {
15     // clientX, clientY는 브라우저가 자동으로 제공하는 이벤트 객체의 속성
16     const {clientX:x, clientY: y} = event
17     dispatch({
18       type, payload:{value}, meta:{x,y}
19     })
20   }
21
22   return(
23     <>
24       <p>Count : {state.count}</p>
25       <button onClick={e => handleClick('INC',1,e)}>+1</button>
26       <button onClick={e => handleClick('DEC',1,e)}>-1</button>
27       <button onClick={e => handleClick('INC',2,e)}>+2</button>
28       <button onClick={e => handleClick('DEC',2,e)}>-2</button>
29     </>
30   )
31
32 }
33
34 export default App

```

countReducer.js

JS

```

1 // 3) countReducer.js
2 export const initialState = {
3   count: 0
4 }
5
6 export function countReducer(state, action) {
7   const { value } = action.payload
8   const { x, y } = action.meta
9   switch (action.type) {
10     case 'INC':
11       console.log(`Click: (${x}, ${y})`)
12       return {
13         ...state,
14         count: state.count + value
15       }
16     case 'DEC':
17       console.log(`Click: (${x}, ${y})`)
18       return {
19         ...state,
20         count: state.count - value
21       }
22     default:
23       throw new Error('Unknown action type')
24   }
25 }

```

결과

Count : 0

+1

-1

+2

-2