Demo

In this lesson, we demonstrate the ToDo app.

WE'LL COVER THE FOLLOWING

Try demo

We simply put together all the code from previous lessons. Run it, and see how it works.

```
import { useReducer } from 'react';
import { createContainer } from 'react-tracked';

const initialState = {
  todos: [
    { id: 1, title: 'Wash dishes' },
    { id: 2, title: 'Study JS' },
    { id: 3, title: 'Buy ticket' },
  ],
  query: '',
};

let nextId = 4;

const reducer = (state, action) => {
  switch (action.type) {
    case 'ADD_TODO':
      return {
        ...state,
        todos: [...state.todos, { id: nextId++, title: action.title }],
      };
    case 'DELETE_TODO':
      return {
        ...state,
        todos: state.todos.filter(todo => todo.id !== action.id),
      };
    case 'TOGGLE_TODO':
      return {
        ...state,
```

```
        todos: state.todos.map(todo =>
          todo.id === action.id
            ? { ...todo, completed: !todo.completed }
            : todo,
        ),
      };
    case 'SET_QUERY':
      return {
        ...state,
        query: action.query,
      };
    default:
      return state;
  }
};

const useValue = () => useReducer(reducer, initialState);

export const {
  Provider,
  useTrackedState,
  useUpdate: useDispatch,
} = createContainer(useValue);
```

How to confirm render optimization

The highlight query is only effective against incomplete ToDo items. This means that complete items do not update with the query change.

You can try it with the following steps.

1. Click checkboxes of some items that will make those items complete. Complete items are shown with a line through them.
2. Type some characters in the query field at the bottom.
3. You will see only incomplete items (not shown with a line through) re-render (become bold).

Next

In the next chapter, we create the same ToDo app in a more structured way.