**Три принципа**

Подведём итог. Что главное в Redux:

* Single source of truth — используя Redux, мы работаем только с одним хранилищем на приложение. Всё состояние в одном месте
* State is read-only — единственный способ изменить состояние – послать действие внутрь хранилища
* Changes are made with pure functions — внутри хранилища можно использовать *только* чистые функции, благодаря этому мы можем "путешествовать во времени"
* Ниже полный пример использования Redux:
* **import** { createStore } **from** 'redux';
* *// Редьюсер – функция, которая описывает то, как изменяются данные внутри хранилища*
* *// Она принимает на вход текущее состояние приложения и должна вернуть новое*
* *// Именно так работает функция reducer, отсюда и название, но оно может быть любым*
* *// Второй параметр описывает действие, с его помощью мы узнаем*
* *// как конкретно надо обновить данные для конкретного вызова*
* *// action — это объект, в котором обязательно есть поле type, содержащее имя действия*
* **const** reducer **=** (state **=** 0, action) **=>** {
* **switch** (action.type) {
* **case** 'INCREMENT':
* **return** state **+** 1;
* **case** 'DECREMENT':
* **return** state **-** 1;
* **default**: *// действие по умолчанию – возврат текущего состояния*
* **return** state;
* }
* };
* *// Создание хранилища на основе редьюсера*
* *// Именно в этом хранилище находится состояние, которое возвращает редьюсер*
* **const** store **=** createStore(reducer);
* *// Состояние можно извлечь с помощью функции getState()*
* store.getState(); *// 0 – так как это начальное значение состояния*
* *// Функция subscribe позволяет подписываться на изменение состояния внутри хранилища*
* *// Она очень похожа на addEventListener, но без указания события*
* *// Как только меняется любая часть состояния, хранилище вызывает переданную функцию*
* *// Здесь мы просто извлекаем состояние и печатаем его на экран*
* store.subscribe(() **=>** console.log(store.getState()));
* *// dispatch – функция, которая вызывает редьюсер*
* *// Редьюсер увеличивает состояние на единицу*
* store.dispatch({ type: 'INCREMENT' }); *// 1*
* *// Редьюсер увеличивает состояние на единицу*
* store.dispatch({ type: 'INCREMENT' }); *// 2*
* *// Редьюсер уменьшает состояние на единицу*
* store.dispatch({ type: 'DECREMENT' }); *// 1*
* store.getState(); *// 1*
* *// Для избежания дублирования и повышения уровня абстракции, вынесем действия в функции*
* **const** increment **=** () **=>** ({ type: 'INCREMENT' });
* **const** decrement **=** () **=>** ({ type: 'DECREMENT' });
* store.dispatch(increment()); *// 2*
* store.dispatch(decrement()); *// 1*