# **06.08.2020 2 часа – Document JS + 4 часа - Redux Отчет: познакомился с Flux и Redux, разобрал их отличия, написал Redux-функцию reducer**

**Redux** – это отличная структура управления состоянием, обычно используется вместе с React.  
В SPA управление данными со стороны клиентской части намного сложнее, чем кажется. В React можно управлять состоянием, но, когда приложение растет, делать становится это неудобно, так как появляются ошибки и ловушки. Facebook придумал решение для такой проблемы – Flux.

**Flux** – паттерн управления состоянием, который дополняет составные компоненты React, используя однонаправленный поток данных. У Flux очень много хранилищ, и каждое хранилище использует разные небольшие части состояния или данных в нашем приложении. То есть каждый отдельный модуль имеет свое хранилище

**Flux data flow** – поток данных Flux  
1) Пользователь взаимодействует с элементом, элемент запускает действие  
2) Действие отправляет соответствующую функцию, затем эта функция изменяет хранилище  
3) Когда хранилище обновляет свои данные, элементы, которые зависят от этого хранилища, обновляются автоматически. Нам не нужно изменять данные в модулях

Это и есть однонаправленный поток данных. Когда приложение становится больше, несколько хранилищ управляют данными

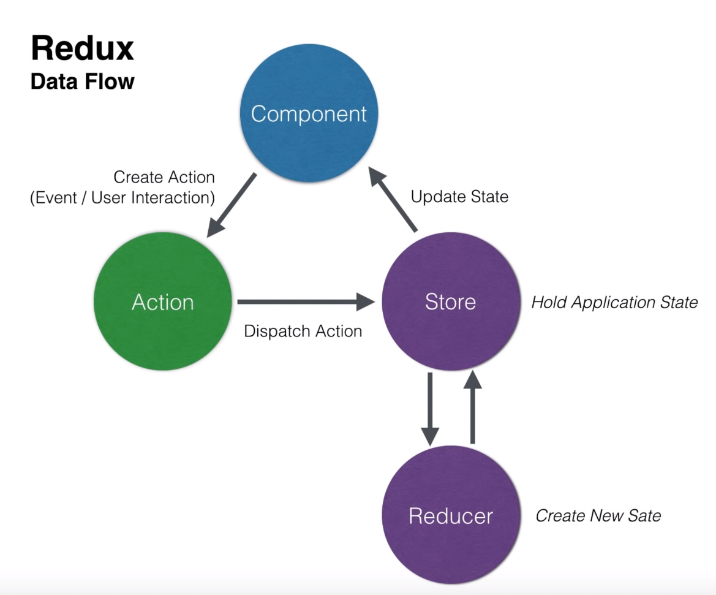
При наличии нескольких хранилищ состояние нашего приложения функционирует по схеме выше, но поток данных является однонаправленным

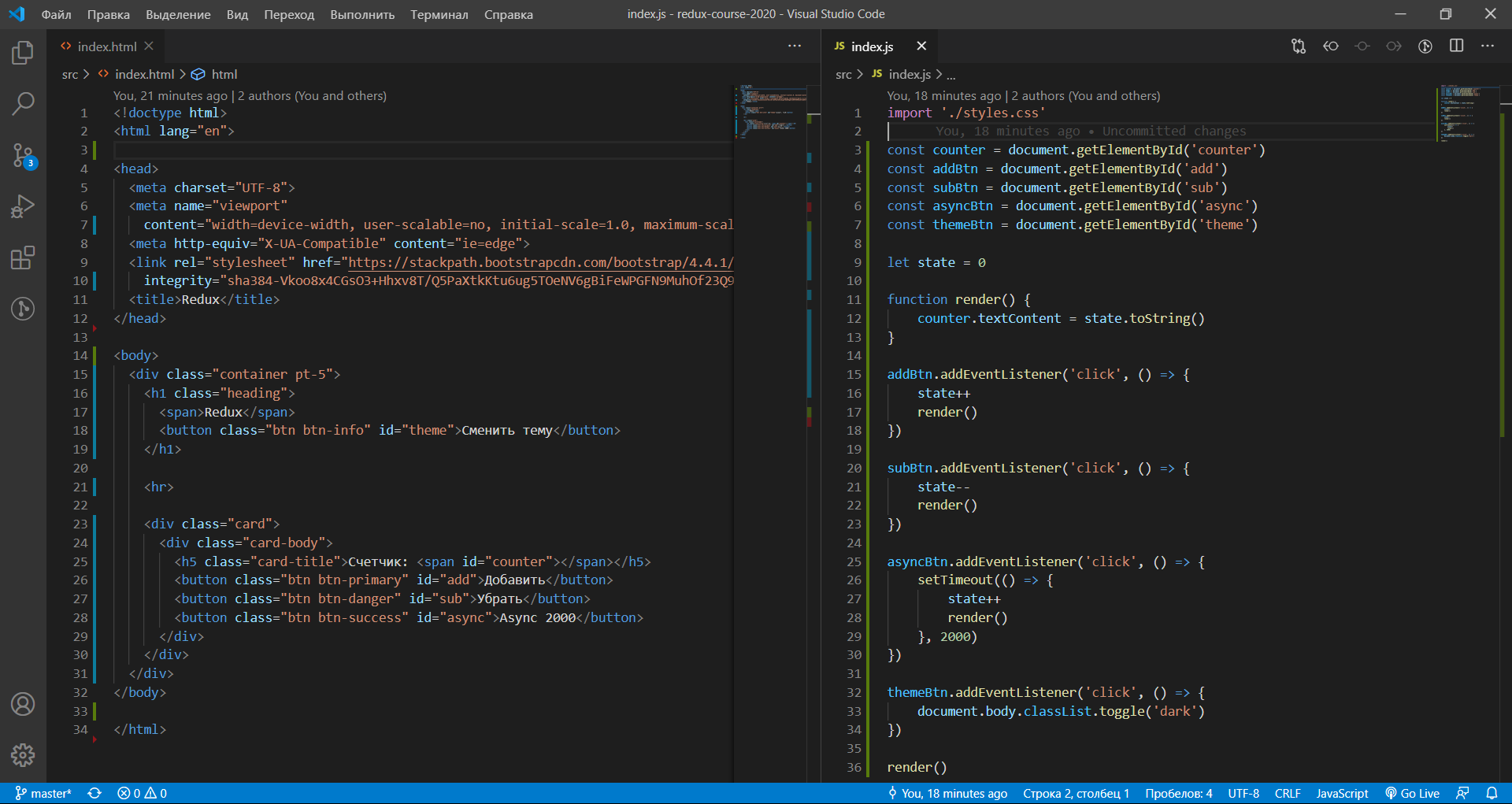
Flux имеет несколько хранилищ

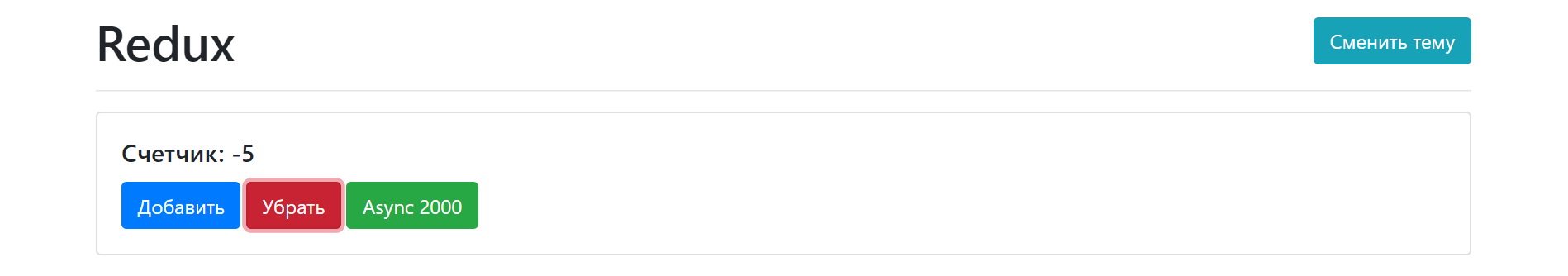
**Redux** – предсказуемый контейнер состояний для приложений JavaScript. Redux также придерживается однонаправленного потока, но он полностью отличается от Flux, т.к Flux может иметь множество потоков

**3 принципа Redux**1) Единственный источник истинны  
2) Состояние доступно только для чтения  
3) Изменения можно делать только с помощью чистых функций

В Redux состояние всего приложения хранится в объекте в одном хранилище  
Единственный способ изменить состояние - вызвать action и объект описывающий, что произошло  
Redux работает независимо от React, его можно использовать в JS, Angular, Vue

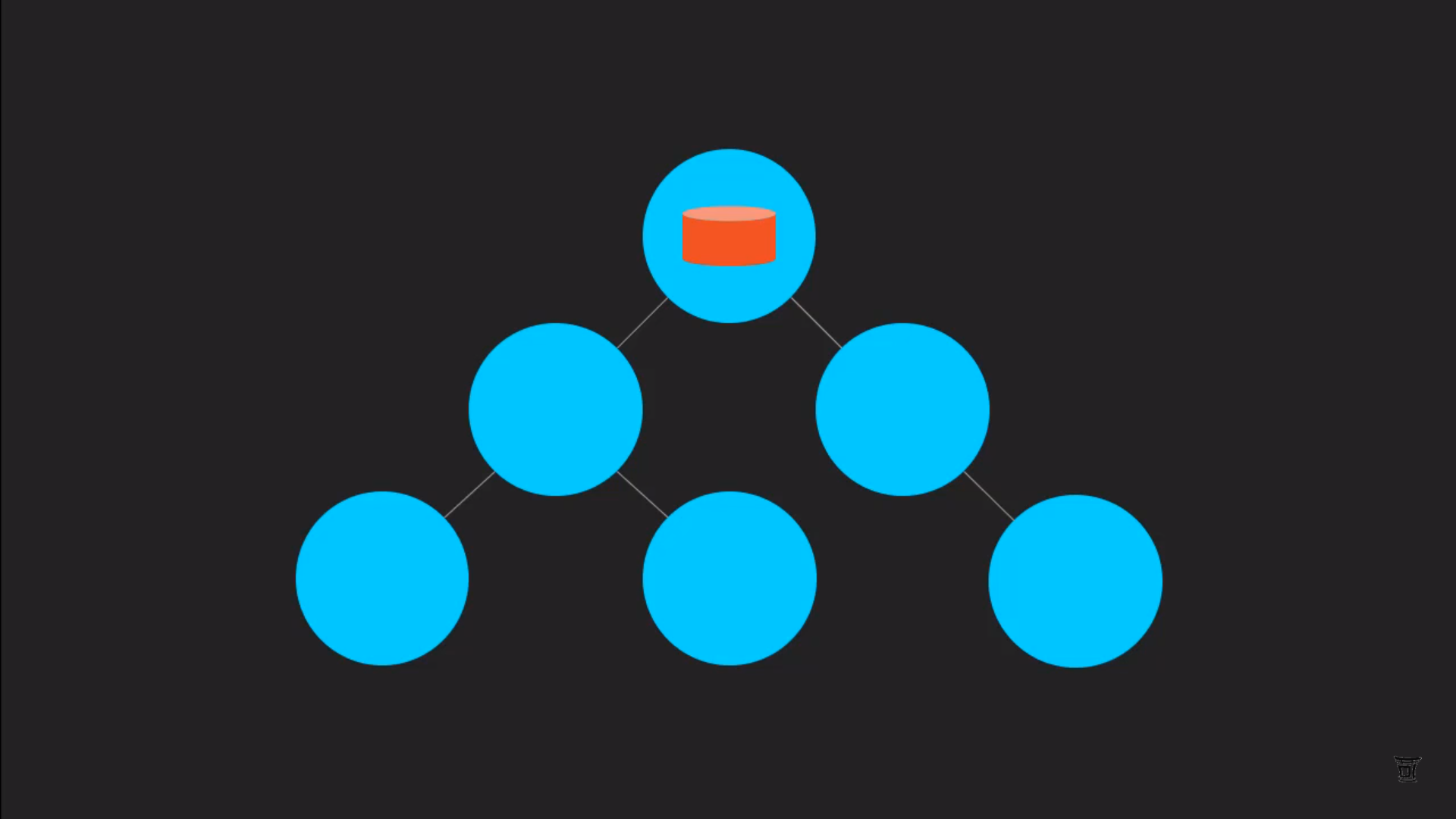
Redux работает по следующей схеме  
  
  
Рассмотрим код ниже, который хранит состояние в переменной state

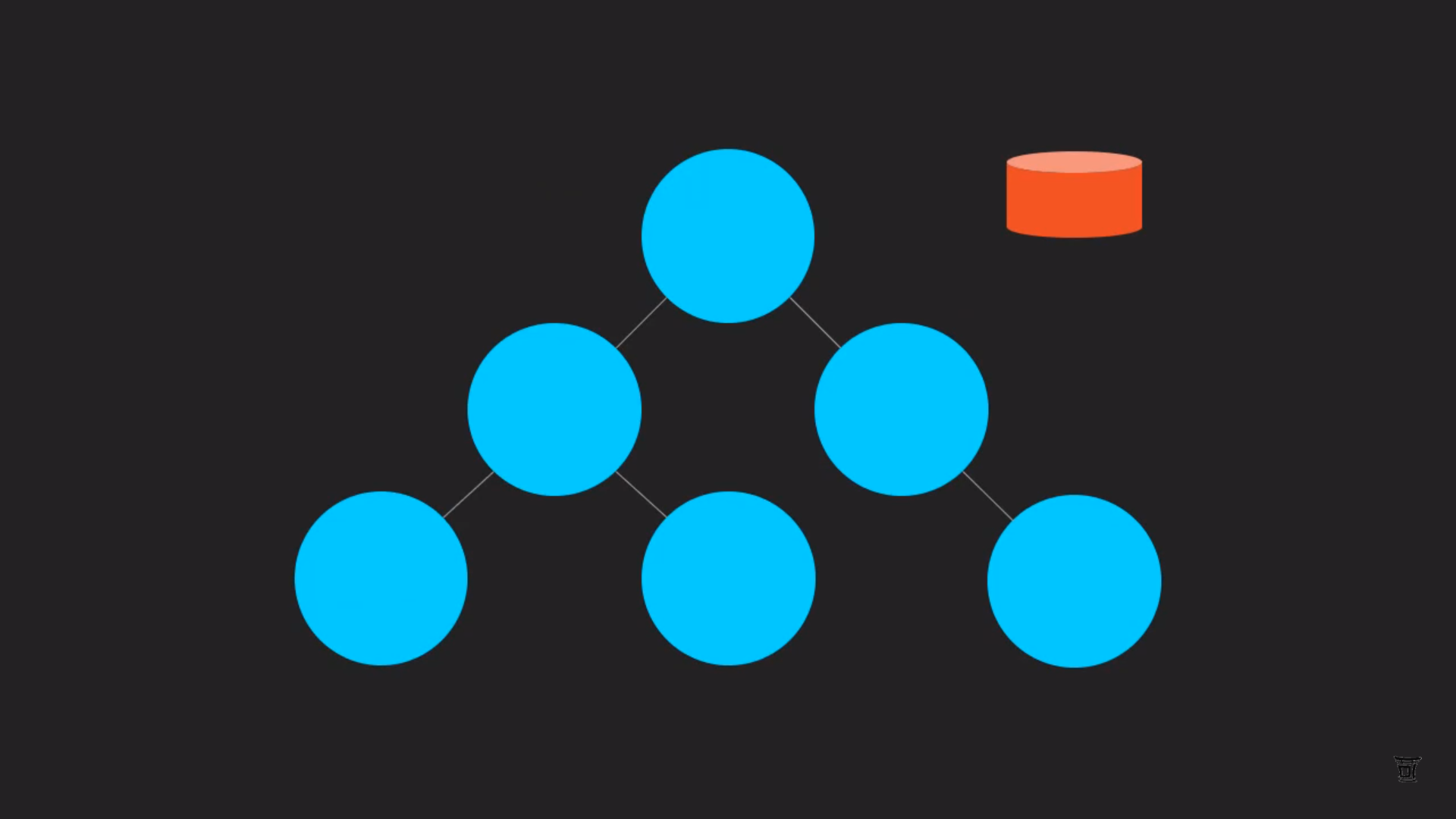


Данное приложение может менять тему, инкрементировать и декрементировать счетчик, а также делать инкремент с асинхронностью 2000 мс

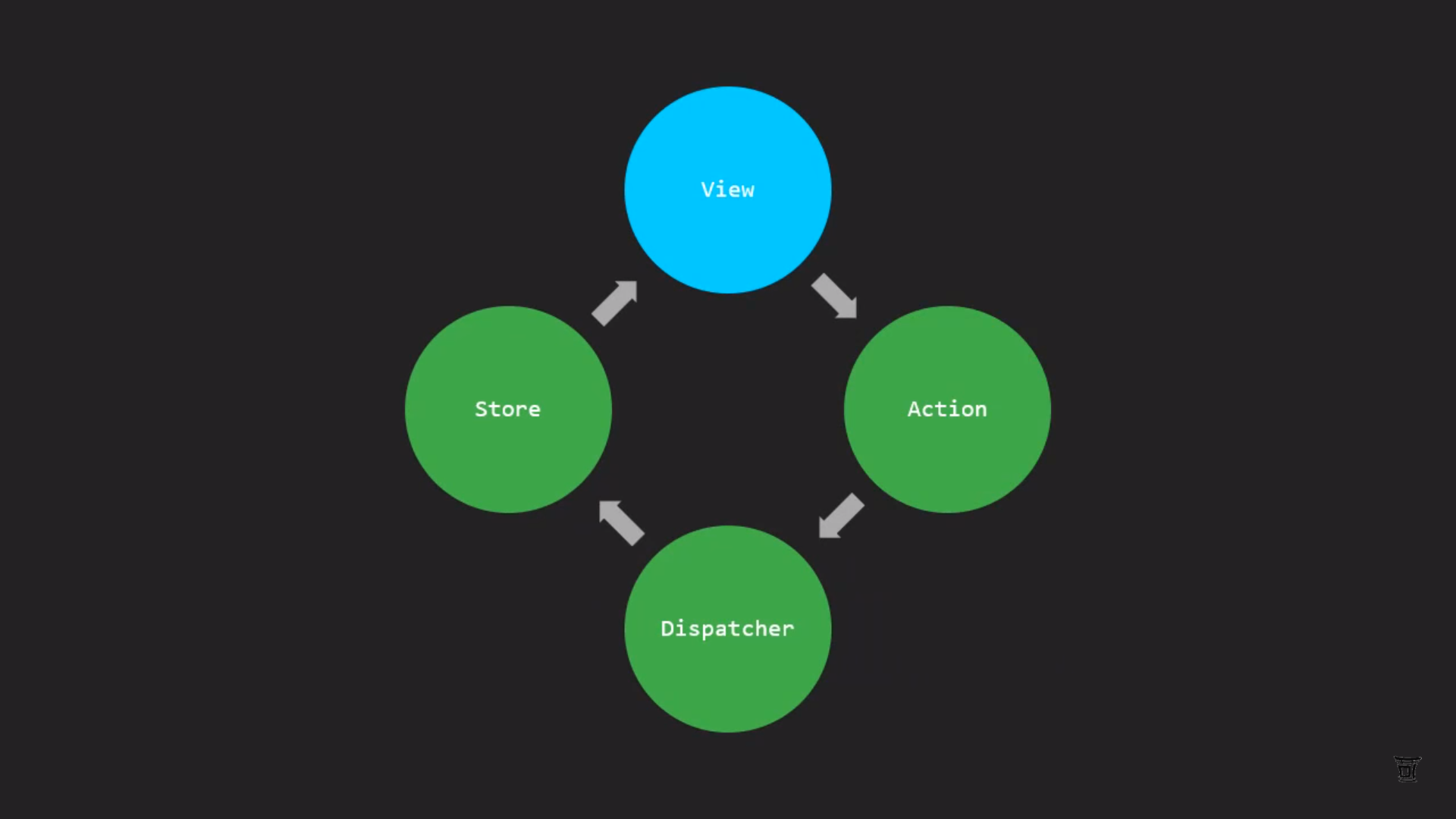
Данное приложение работает достаточно хорошо, но у него есть проблема: когда мы обновляем страницу, наши данные теряются, то есть счетчик принимает начальное значение 0, а тема становится стандартного цвета

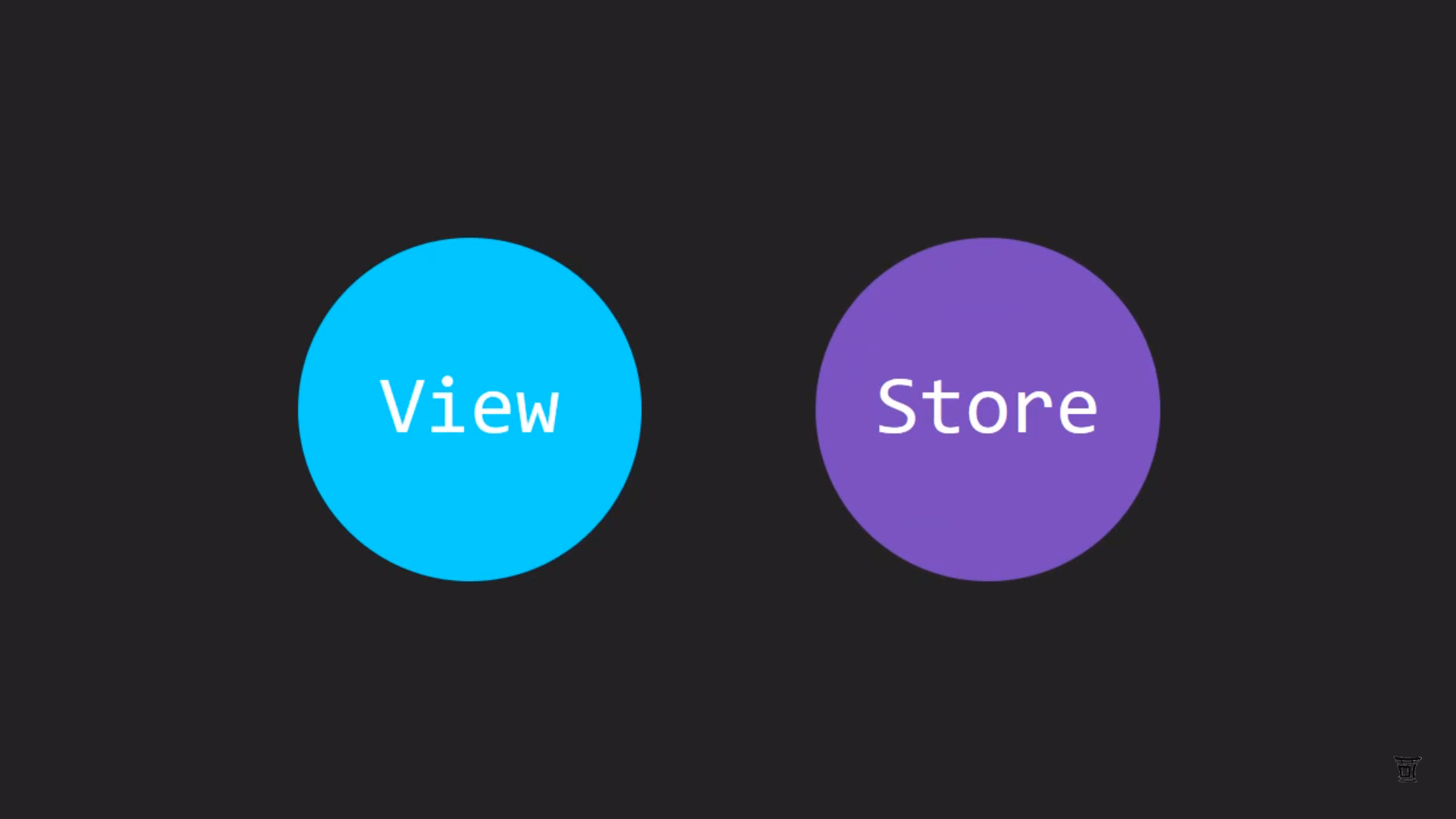
**Введение в Flux и Redux**

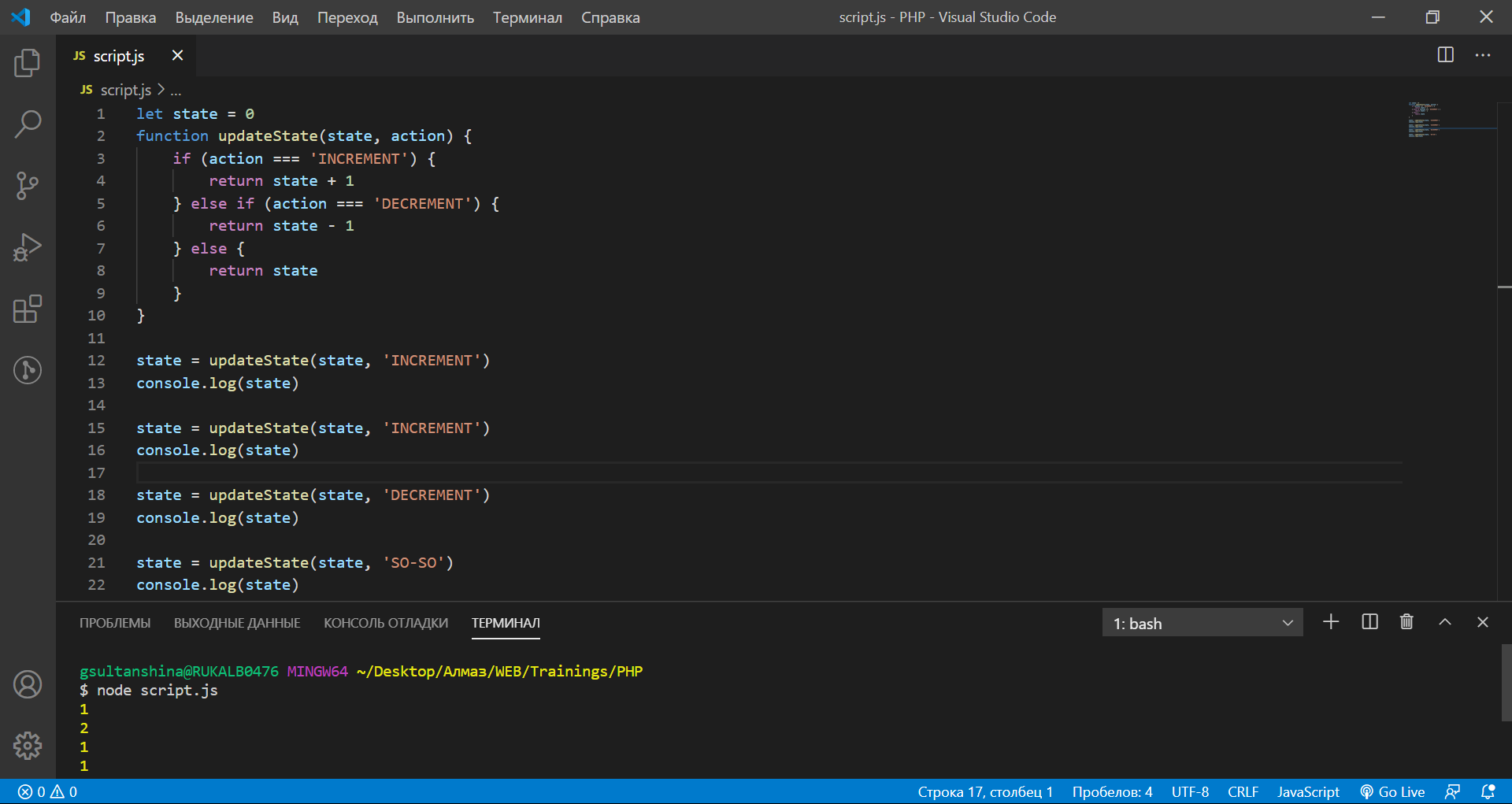
В React состояние рекомендуется размещать как можно выше, в главном компоненте  
При такой архитектуре данные спускаются вниз по свойствам  
Для изменения состояния компоненты сообщают своим родителям об обновлении состояния с помощью передаваемых функций  
Далее основной компонент обновляет состояние и вновь отправляет данные вниз  
Для многих приложений это является нормальным, но с расширением приложения появляется больше компонент, и такая архитектура является не очень удобной

Разработчики Facebook, столкнувшись с такой проблемой, придумали новую архитектуру Flux  
Главной идеей Flux является деление состояния от компонентов. Таким образом любой компонент может взаимодействовать с состоянием напрямую

Flux состоит из 4 частей:  
1) view – пользовательский интерфейс (в React – это компонента)  
2) store – хранилище. Здесь находится состояние приложения  
3) dispatcher – диспетчер. Сообщает хранилищу о каком то событии, передает необходимую инфу  
4) action – действие или событие, которое происходит в приложении

Если в общем, то компонента генерирует действие, диспетчер сообщает об этом хранилищу, хранилище изменяет состояние, данные передаются в компоненту  


Flux имеет несколько реализаций, но самой популярной из них является Redux. Главной характерной чертой Redux является простота. В нем компонента взаимодействует с хранилищем напрямую. С помощью action компонента сообщает хранилищу, что произошло в приложении. Также в Redux имеется dispatcher, но он является частью хранилища  


**Пишем функцию Reducer (преобразователь)**Данная функция не обновляет исходное состояние, она возвращает новое состояние****