

39.

Жизненный цикл компонентов

Цель:

Познакомиться со следующими понятиями и возможностями react:

- жизненный цикл компонента

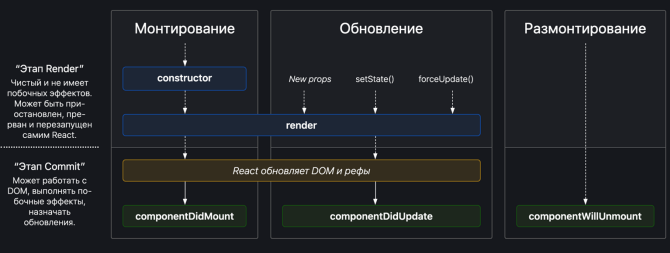
- HOC (Higher-Order Components)

- умные/глупые (smart/dumb) компоненты

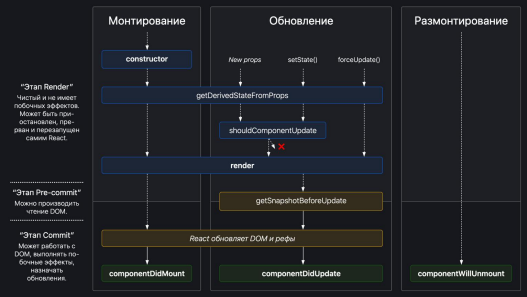
- Container и Presentational Components

- композиция компонентов

Жизненный цикл компонента:



Жизненный цикл компонента:



Монтирование:

При создании экземпляра компонента и его вставке в DOM, следующие методы вызываются в установленном порядке:

**1. constructor()**

2. static getDerivedStateFromProps()

**3. render()**

**4. componentDidMount()**

Обновление:

Обновление происходит при изменении пропсов или состояния. Следующие методы вызываются в установленном порядке при повторном рендере компонента:

1. static getDerivedStateFromProps()

2. shouldComponentUpdate()

**3. render()**

4. getSnapshotBeforeUpdate()

**5. componentDidUpdate()**

Размонтирование:

Вызывается при удалении компонента из DOM:

**componentWillUnmount()**

Каждый раз когда DOM-узел, созданный компонентом, удаляется, происходит «размонтирование» (unmounting). Чтобы избежать утечки ресурсов, необходимо не забывать делать очистку таймеров, прослушивателей событий и других эффектов, когда мы удаляем компоненты.

Обработка ошибок:

Следующие методы вызываются, если произошла ошибка в процессе рендеринга, методе жизненного цикла или конструкторе любого дочернего компонента.

● static getDerivedStateFromError()

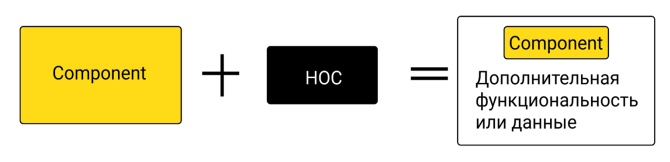
● componentDidCatch()

HOC (Higher-Order Components):

**HOC -** это паттерн, который позволяет переиспользовать функционал между компонентами без повторения кода.

**HOC -** представляет собой функцию, которая принимает компонент и возвращает новый компонент. Такая функция позволяет добавлять дополнительные данные в компонент либо функционал.

HOC:

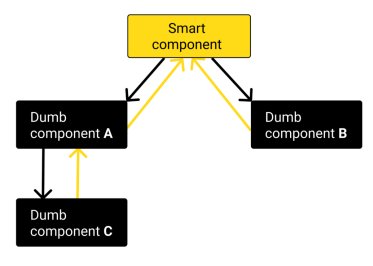


HOC:

● HOC должны быть чистыми функциями без сайд эффектов. ● Нельзя использовать HOC в рендере компонента. Получать доступ к HOC необходимо только за пределами определения компонента.

● Статические методы должны быть скопированы, чтобы к ним оставался доступ. Т.к. при применении HOC, мы заворачиваем оригинальный компонент в контейнер, и у нового компонента не остается статических методов оригинального компонента.

● Рефы(Ref) не передаются. HOC передает все props оборачиваему компоненту кроме ref.

Умные/глупые(Smart/Dumb) компоненты:

Dumb Components:

**Компоненты Dumb (глупые компоненты)** также называются «презентационными компонентами», потому что они отвечают только за представление DOM.

Такой компонент содержит в себе только один метод render() и не содержит никаких изменяемых состояний.

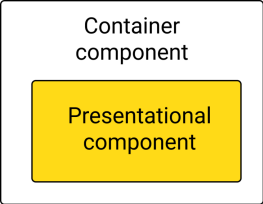
Smart Components:

● хранит состояние стора и пробрасывает его как объекты в глупые компоненты

● вызывает Flux actions и обеспечивает ими глупые компоненты в виде колбэков

● редко владеют собственными стилями и хранят в себе дом, для этого используют глупые компоненты

Container и Presentational Components:

● **Container (Smart) —** компонент, в 

котором принимают и работают с

данными

● **Presentational (dumb) —** компонент,

в котором отображают данные.

Container и Presentational Components:

**Польза от данного подхода:**

● Лучшее разделение ответственности.

● Лучшая реюзабельность компонентов. Один и тот же компонент может использоваться с разными источниками данных.