# Важно

Перед оптимизацией нужно сделать замеры и понять, а нужно ли вообще ее делать.

# Этапы обновления компонента

1. **Render phase** — формирование текущего дерева
2. **Reconciliation** — сравнение с предыдущим деревом и нахождение отличий
3. **Commit phase** — обновление DOM

# Компонент перерендрится:

- При изменении пропсов
- При изменении состояния

# Убираем общее состояние

Пример

```
01. const App = () => {
02.   const [counter, setCounter] = useState(0)
03.
04.   return (
05.     <>
06.       <button onClick={increaseCounter}>{counter}</button>
07.       <ExpensiveTree />
08.     </>
09. )}
```

# Убираем общее состояние

```
01. const Counter = () => {
02.   const [count, setCounter] = useState(0)
03.   return <button onClick={increaseCounter}>{counter}</button>
04. }
05.
06. const App = () => (
07.   <>
08.     <Counter>
09.     <ExpensiveTree />
10.   </>
11. )
```

# Используем children

Пример

```
01. const App = () => {
02.   const [colSize, setColSize] = useState(6)
03.
04.   return (
05.     <div style={{ "--colSize": colSize }} >
06.       <button onClick={increaseColSize}></button>
07.       <ExpensiveTree />
08.     </div>
09. )}
```

# Используем children

```
01. const DynamicCol = ({ children }) => {
02.   const [colSize, setColSize] = useState(6)
03.   return (
04.     <div style={{ "--colSize": colSize }} >
05.       <button onClick={increaseColSize}></button>
06.       {children}
07.     </div>
08. )}
09.
10. const App = () => <DynamicCol><ExpensiveTree /></DynamicCol>
```

# memo

**Не забываем использовать useCallback, если передаётся функция как пропс**

Пример

```
01. const memoizedComponent = memo(Component, areEqual)
02.
03. function areEqual = (prevProps, nextProps) {
04.   if (prevProps.name === nextProps.name) return true
05. }
```

# Оптимизация не поможет, если компонент размонтируется

# Компонент размонтируется:

**Если изменится тип родителя**

Пример

```
01. const Wrapper = isActive ? ActiveWrapper : DefaultWrapper
02.
03. <Wrapper>
04.   <Child />
05. </Wrapper>
```

# Компонент размонтируется:

**Если компонент переместится в родителе и не имеет пропса key**

Пример

```
<>
  <Child />
</>

<>
  <Btn />
  <Child />
</>
```

# Компонент размонтируется:

**Если изменится пропс key**

Пример

```
<>
  <Child key="id" />
</>

<>
  <Child key="another-id" />
</>
```

# Полезные ссылки

- Айти Синяк: React Reconciliation
- Айти Синяк: React key
- Dan Abramov: Before You memo