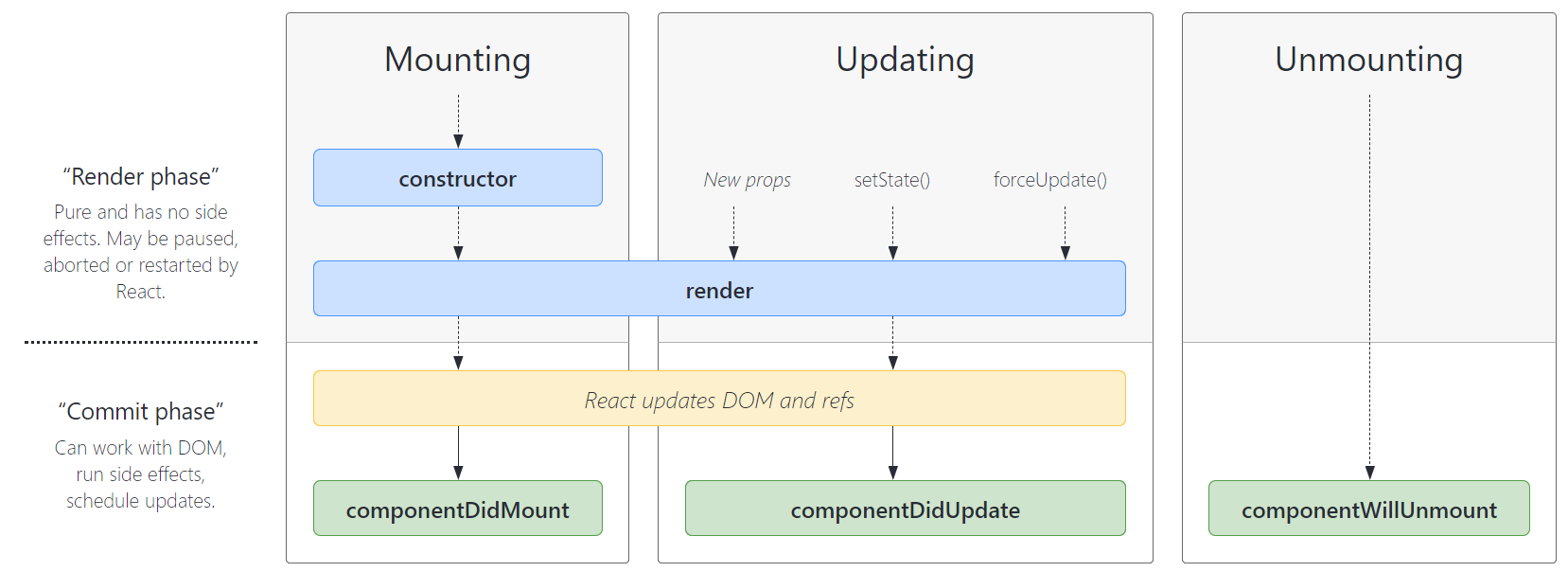
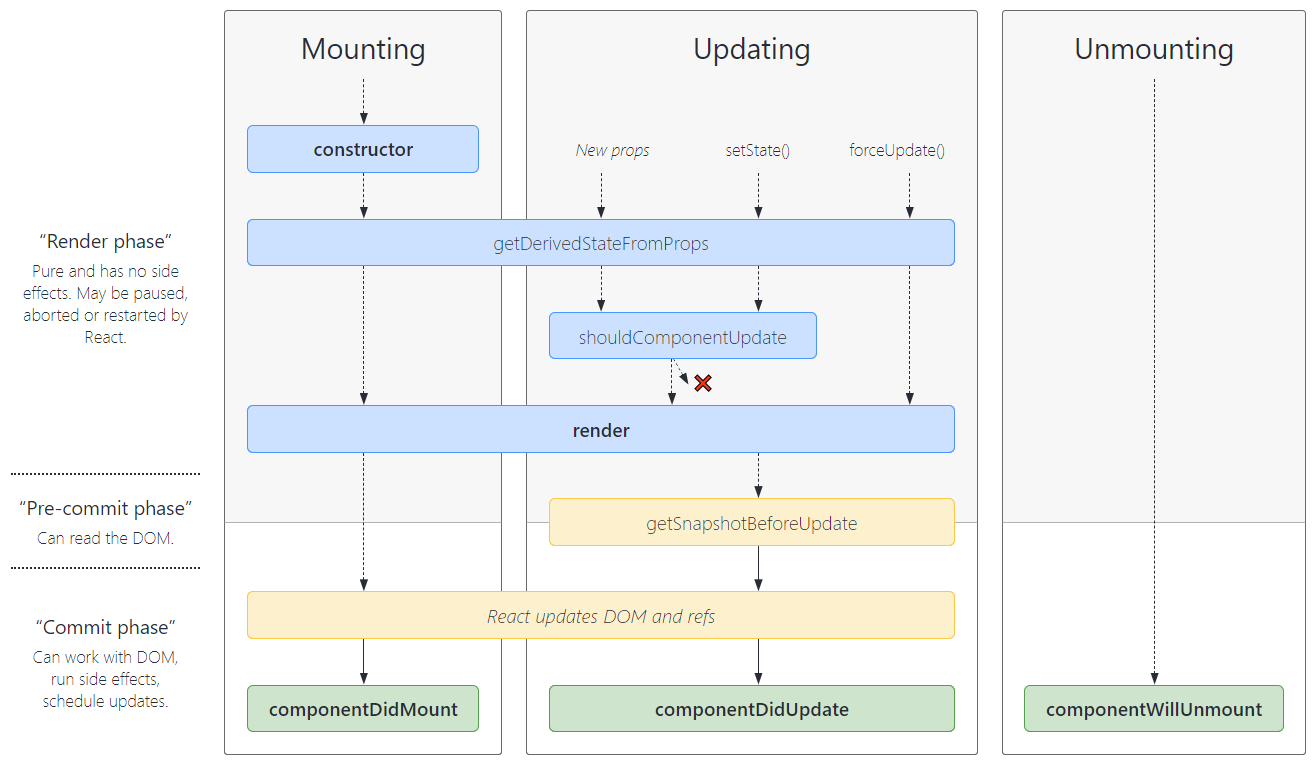
* Mỗi 1 component đều có 1 lifecycle riêng
* **LifeCycle**



*Rút gọn*

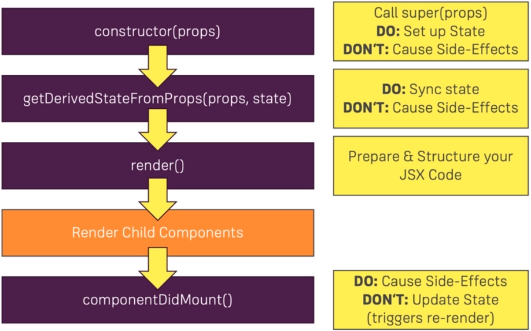


*Đẩy đủ*

[*http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/*](http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/)

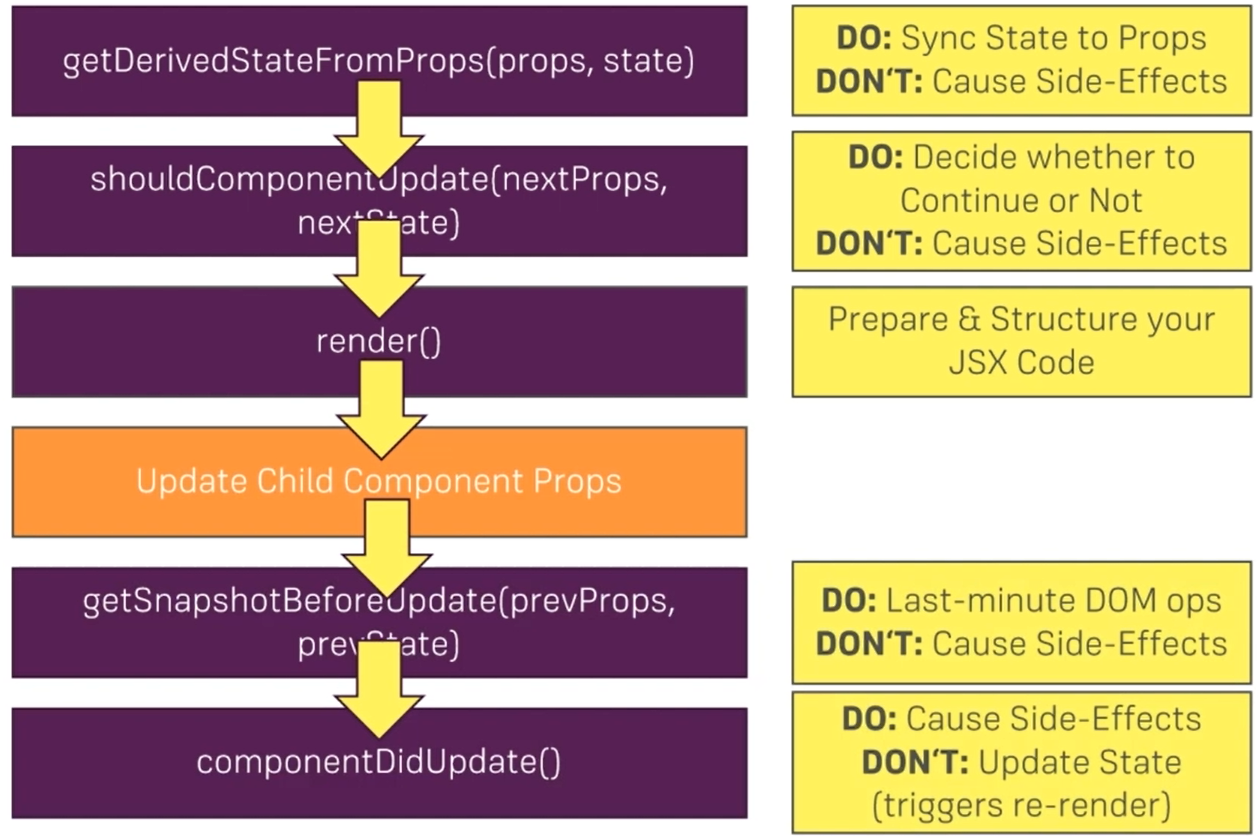
* Mount nghĩa là component render vào trong DOM
  + **Mounting**

Được gọi khi nó render component trong lần đầu tiên

****

* + **Updating** (update props hoặc state)

Được gọi khi có sự thay đổi của props hoặc state

****

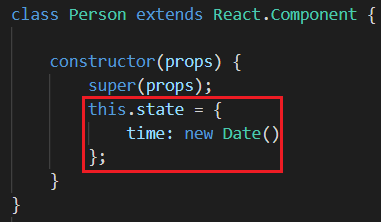
* + **Unmounting**

Được gọi trước khi remove component khỏi DOM

* **constructor()**
  + Sử dụng để
    - Khởi tạo state

Sử dụng trực tiếp this.state, không gọi setState() trong constructor

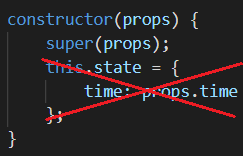
Nếu không khởi tạo state thì không nên gọi constructor để làm gì



* + - Binding method (có thể register trong componentDidMount()) (thay thế bằng Arrow Function)
  + Cần gọi super(props) trước bất kì câu lệnh nào

Nếu không gọi super(props) thì khi sử dụng this.props sẽ bị undefined

* + Tránh copy props vào trong state



* + Trong mỗi component, không nhất thiết phải có constructor (chỉ viết ra khi muốn khởi tạo state)
* static **getDerivedStateFromProps(props, state)**

**Ít khi sử dụng**



* + Method được gọi trước khi render()
  + Nên return ra 1 Object để update state và return null nếu không update
  + Chỉ sử dụng khi value của state phụ thuộc vào prop (hiếm khi sử dụng)

<https://reactjs.org/docs/react-component.html#static-getderivedstatefromprops>

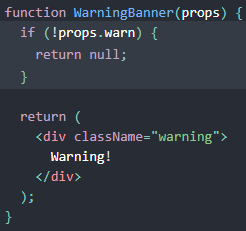
* **render()**

**required**

render() không được gọi nếu shouldComponentUpdate() return false

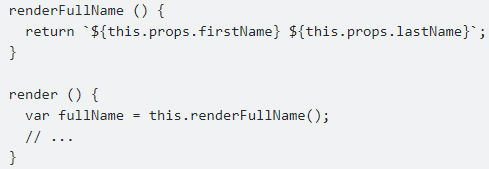
****

* + Thông qua JSX để render ra DOM node
    - Không return ra boolean và null
    - Nếu return ra null thì component sẽ không hiển thị



[*https://codepen.io/gaearon/pen/Xjoqwm?editors=0010*](https://codepen.io/gaearon/pen/Xjoqwm?editors=0010)

* + Dựa vào this.props và this.state
  + Phải return ra 1 trong những cái sau
    - React elements
    - Arrays and fragments
    - Portals
    - String and numbers
    - Booleans or null
  + Ngoài JSX thì nó sẽ render cả child component, khi render hết tất cả child component thì sẽ gọi method componentDidMount() để kết thúc
  + Không nên tính toán hay so sánh gì trong method này



* + Không nên gọi API hoặc setTimeout() trong phương thức này
  + Không nên chứa quá nhiều jsx (sẽ render lại nhiều khi state thay đổi) 🡺 với list thì tạo riêng 1 component chứa list
* **componentDidMount()**



* + Method này được gọi ngay khi 1 component is mounted (component đã được render vào trong DOM 🡺 component đã hiện hình)
  + Nên có thể
    - init các DOM
    - load data (call API)
    - subscription event (unsubscript ở componentWillUnmount())
  + Có thể setState() trong method này nhưng sẽ bị render() sẽ bị 2 lần 🡺 dẫn tới bad performance, vẫn nên init state trong constructor() (trừ khi sử dụng modal, tooltip cần tính toán vị trí của DOM trước khi render hoặc gọi Promise)
* **shouldComponentUpdate()**

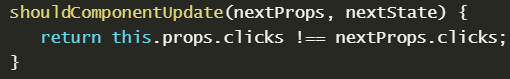
**Ít khi sử dụng**

****

*default: return true*

* + Cho phép tính toán khi nào nên render lại
  + Return true thì method render() và các method liên quan tới updating sẽ được gọi
  + Được gọi trước khi component nhận new props, state
  + Method này không được gọi khi init render, forceUpdate()

🡺 Cải thiện performance vì không phải lúc nào cũng muốn update

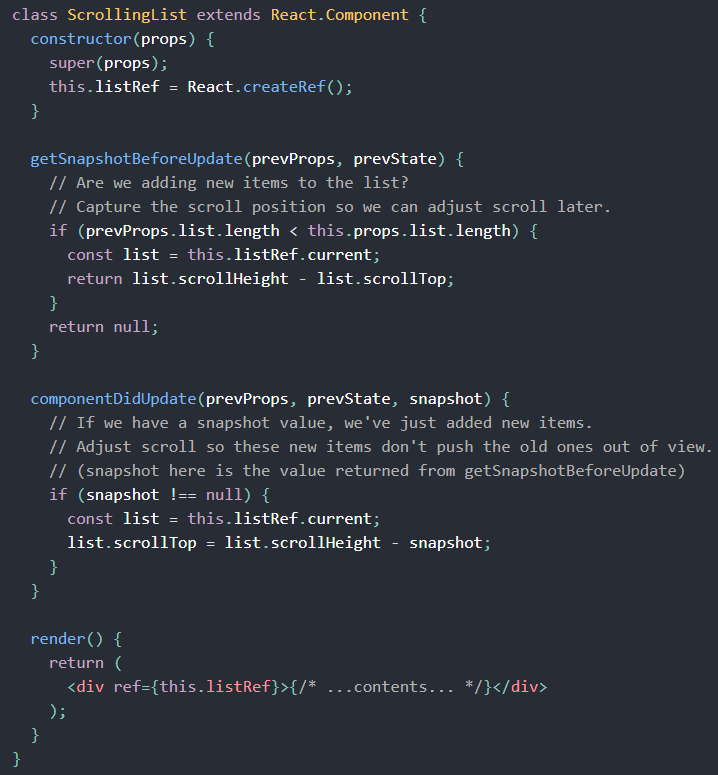


* **getSnapshotBeforeUpdate()**

**Ít khi sử dụng**

****

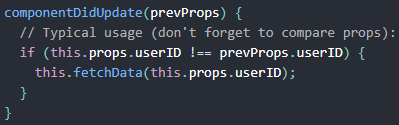
* + Gọi trước khi render xuống DOM
  + Cho phép lấy thông tin của DOM như: vị trí thanh scroll, …
  + Sẽ return ra value (snapshot) và sẽ đẩy xuống method componentDidUpdate()



* **componentDidUpdate()**

****

* + Được gọi khi shouldComponentUpdate() return true
  + Gọi ngay sau khi component được update (không gọi trong lần render đầu tiên)
  + Có thể
    - call network request nếu thấy prop hiện tại khác với previous prop
    - Nếu gọi setState() thì phải đưa nó trong câu điều kiện, nếu không sẽ bị loop
  + Nếu implement phương thức getSnapshotBeforeUpdate() thì giá trị return của sẽ được đưa vào snapshot, nếu không thì là undefined



* **componentWillUnmount()**



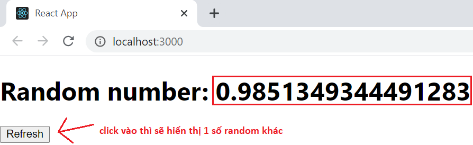
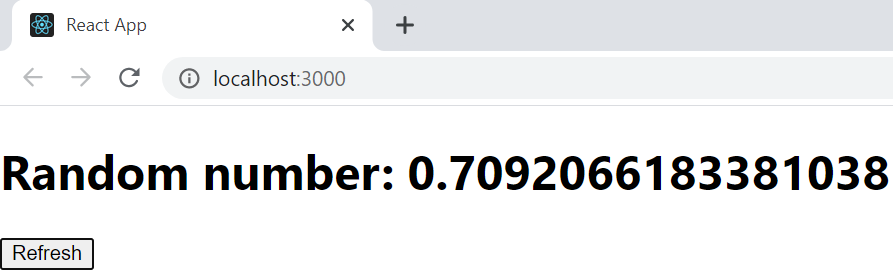
* + Có thể
    - Remove listener
    - Cancel network requests
  + Không được gọi setState() bởi vì component sẽ không render lại
* **forceUpdate()**

Re-render lại component (bỏ qua method shouldComponentUpdate() của component gọi)



VD:

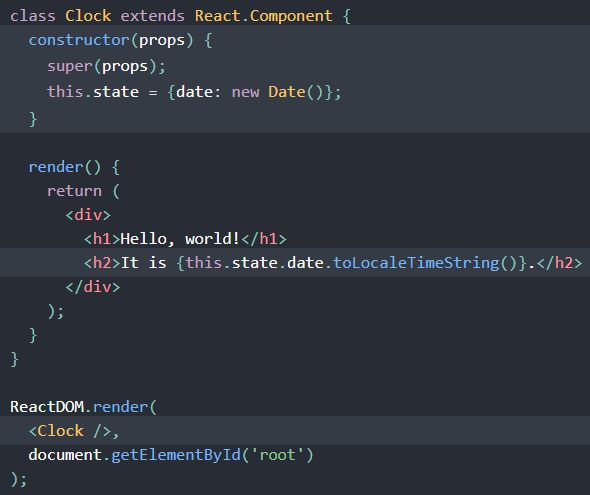
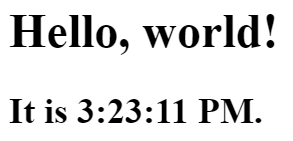


* **Old version** (không nên sử dụng)
  + **componentWillMount()**
  + **componentWillRecieveProps()**
  + **componentWillUpdate()**

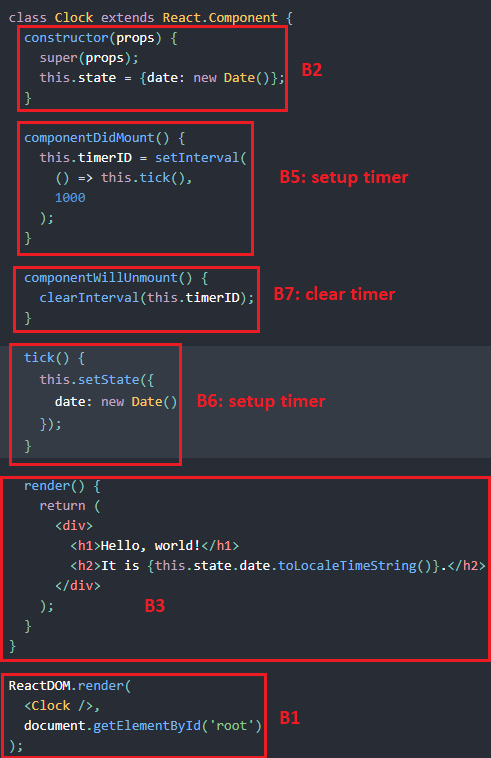
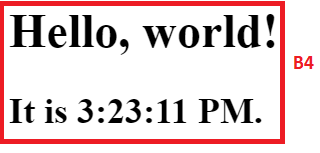
<https://reactjs.org/blog/2018/03/27/update-on-async-rendering.html>

* VD1:

*Render component to DOM*

[*https://codepen.io/gaearon/pen/KgQpJd?editors=0010*](https://codepen.io/gaearon/pen/KgQpJd?editors=0010)

* *

*Add lifecycle method*

[*https://codepen.io/gaearon/pen/amqdNA?editors=0010*](https://codepen.io/gaearon/pen/amqdNA?editors=0010)

**Flow**

* B1: <Clock> được truyền vào ReactDOM.render()
* B2: React gọi constructor của Clock, khởi tạo state, props
* B3: Sau đó React gọi method render() của <Clock>
* B4: output của method render() sẽ được React insert vào DOM và screen để hiển thị time hiện tại lên màn hình
* B5: React gọi tới method componentDidMount(). Ta sẽ setup timer tại method này
* B6: Mỗi 1 giây thì ta sẽ gọi setState() 1 lần (thông qua method tick()) 🡺 React sẽ update lại UI bằng cách gọi lại method render()
* B7: Khi <Clock> Component bị remove khỏi DOM thì sẽ gọi method componentWillUnmount(). Ta sẽ clear timer ở đây

*Đọc thêm:* [*https://reactjs.org/docs/state-and-lifecycle.html*](https://reactjs.org/docs/state-and-lifecycle.html)

VD2:

|  |  |  |
| --- | --- | --- |
|  | *Mounting* | *Updating* |

* **Reference**

<https://reactjs.org/docs/state-and-lifecycle.html>

<https://reactjs.org/docs/react-component.html>

<https://www.javatpoint.com/react-component-life-cycle>

<https://reactjs.org/docs/error-boundaries.html>