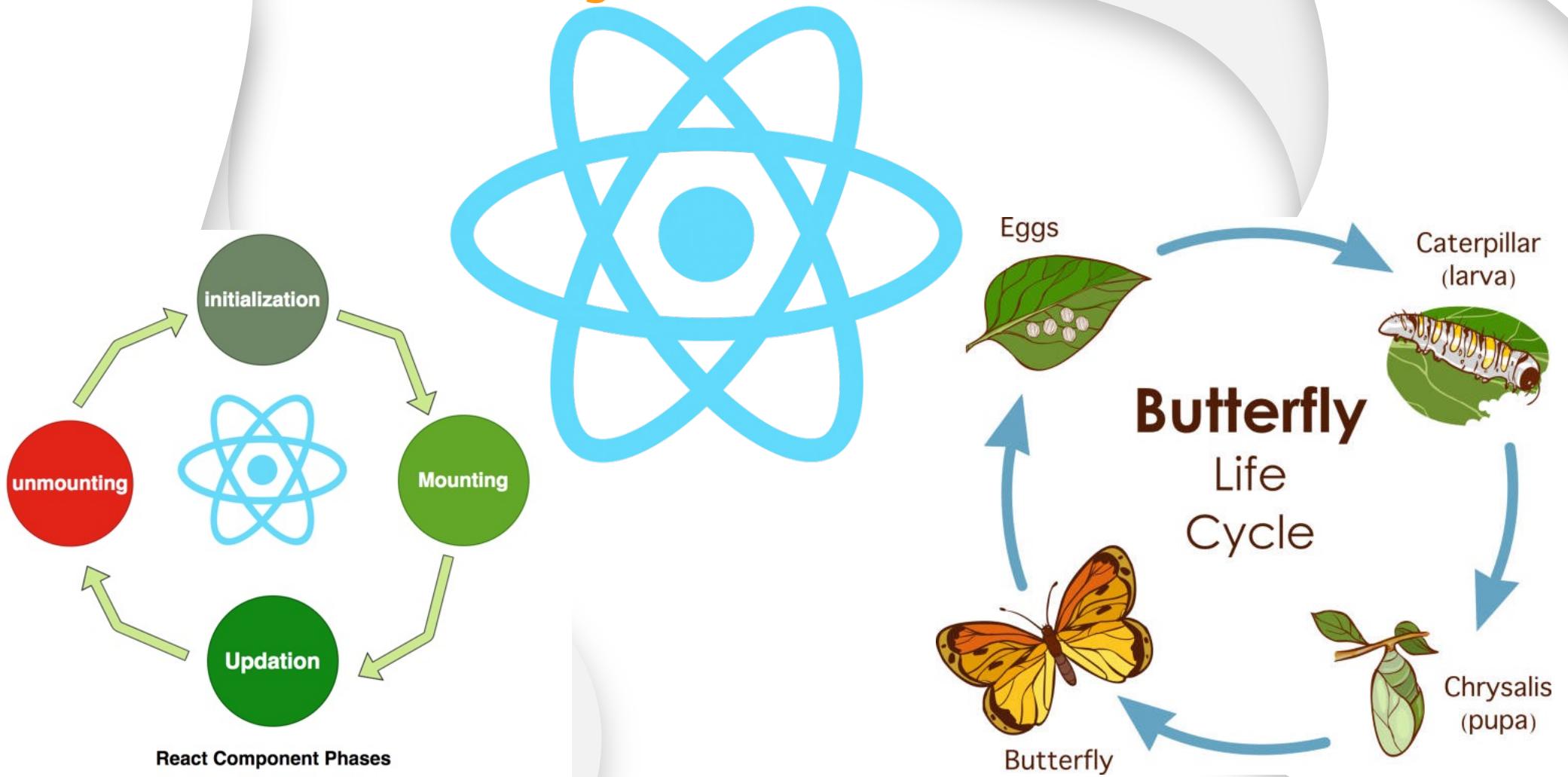
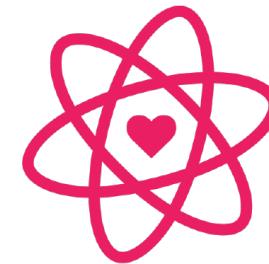


React js

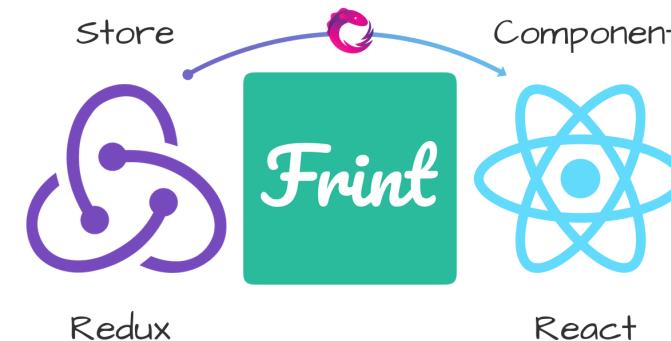
Life cycle - axios



Giới thiệu

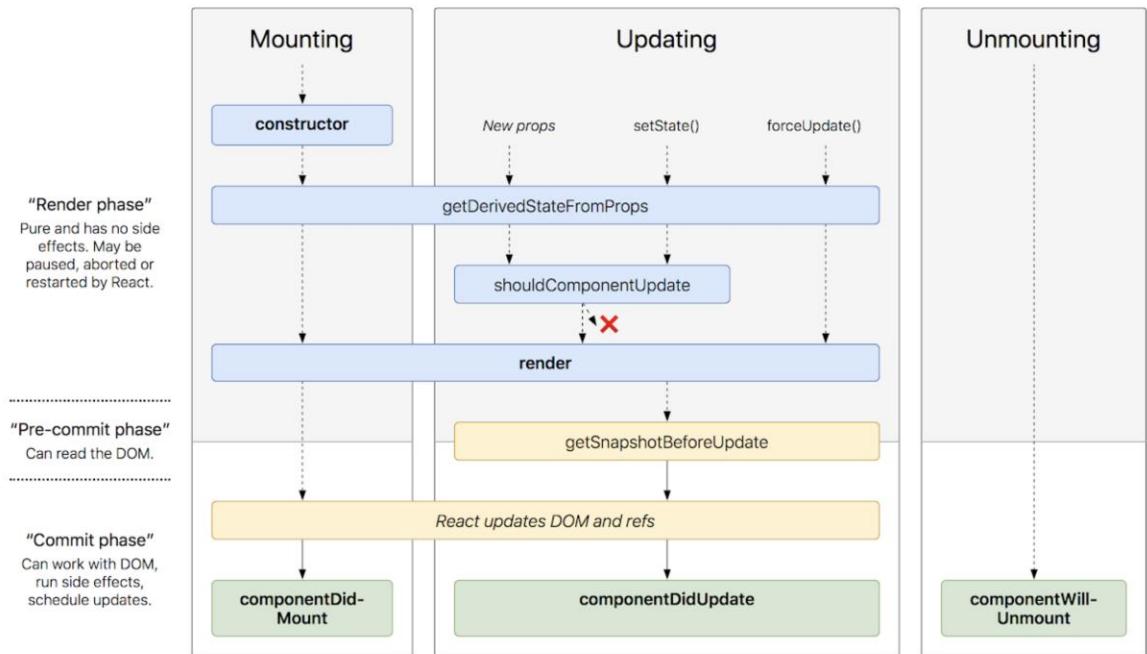
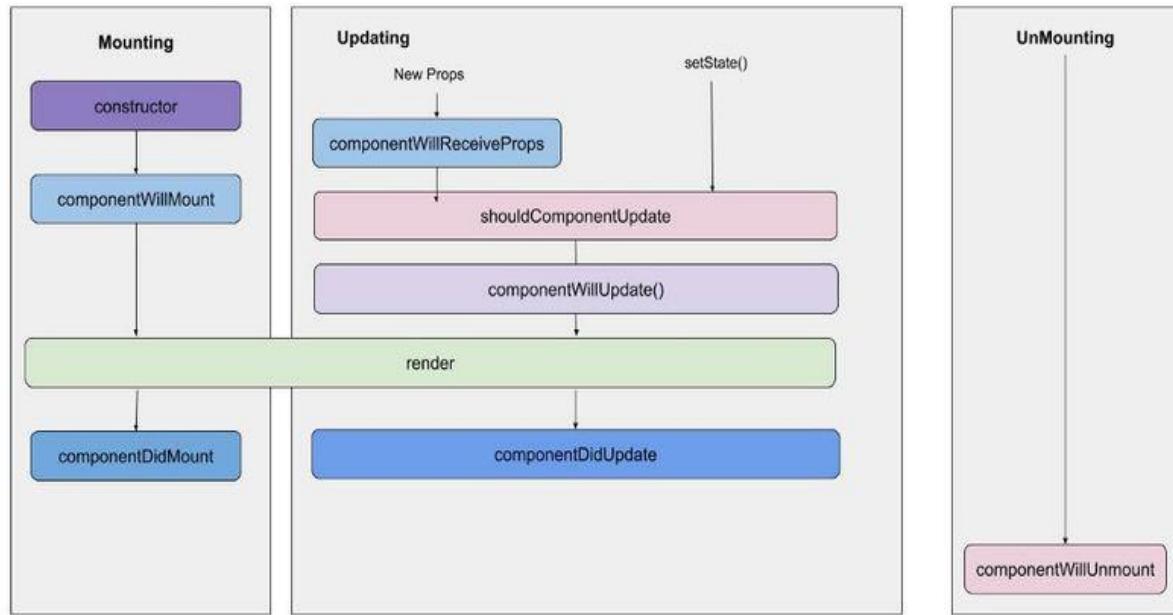


React JS



□ Life-cycle react

React 16.0 Life Cycle



React phiên bản 16.0

React phiên bản từ 16.4+ trở đi

Ngoài ra còn bổ sung thêm 1 số lifecycle nữa

❑ Lifecycle react (mounting)

Lifecycle react là thứ tự chạy các phương thức có sẵn của react class component.

Trong đó ta đi vào trạng thái đầu tiên là mounting (Các phương thức lifecycle sẽ được tự động gọi theo thứ tự)

✓ Initialization

Constructor()

Hàm được thực thi ngay khi component được hiển thị lên giao diện. Nơi thường để khởi tạo thuộc tính state và props.

✓ Mounting (React version <16.4)

componentWillMount()

Hàm được thực thi ngay khi component được hiển thị lên giao diện. Nơi thường để khởi tạo thuộc tính state và props.

render()

Hàm render dữ liệu ra giao diện.

componentDidMount()

Hàm thực thi sau hàm render. Thường dùng để các hàm ajax DOM, hoặc update state khi được thực thi

✓ Mounting (React version >16.4)

getDerivedStateFromProps()

Ở phiên bản react 16.4 trở đi `componentWillMount` bị loại bỏ thay thế bằng `getDerivedStateFromProps(nextProps, currentState)`.
Hàm này không chỉ thực thi ở mounting còn thực thi ở updating khi state hoặc props thay đổi

render()

Hàm render dữ liệu ra giao diện.

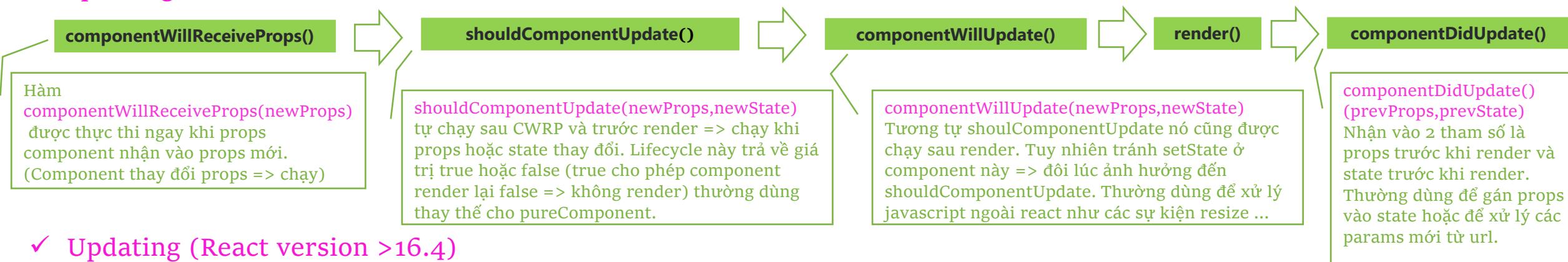
componentDidMount()

Hàm thực thi sau hàm render. Thường dùng để các hàm ajax DOM, hoặc update state khi được thực thi

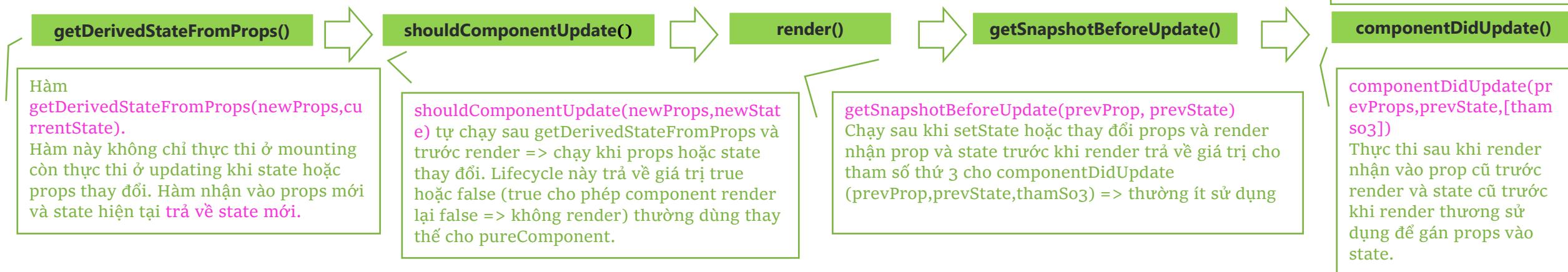
❑ Lifecycle react (updating)

- Tương tự như mounting các hàm ở lifecycle updating cũng được thực thi theo tuần tự. Nhưng ứng với sự thay đổi bởi thuộc tính state hoặc props của component.
- state thay đổi khi ta `setState` lifecycle này sẽ chạy, props thay đổi khi giá trị truyền vào từ component cha (props) hoặc redux (mapStateToProps) thay đổi thì lifecycle này chạy

✓ Updating (React version <16.4)



✓ Updating (React version >16.4)



❑ Lifecycle react (unmounting)

Lifecycle unmounting là các hàm khi component đó biến mất khỏi giao diện (Remove khỏi DOM) sẽ thực thi.

- ✓ unmounting

`componentWillUnmount()`

`componentWillUnmount()` :

Life cycle duy nhất trong unmounting

componentWillUnmount sẽ được gọi trước khi một component bị remove khỏi một DOM. Nếu một component khởi tạo bất kì một method nào mà method đó yêu cầu phải clean up thì componentWillMount sẽ là nơi bạn nên đặt clean up.

Lifecycle mounting

```
Lifecycle.jsx
import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
import Child from './Child';
export default class Lifecycle extends Component {
  constructor(props) {
    super(props);
    this.state = {
      number:1
    }
    console.log('constructor');
  }
  shouldComponentUpdate (newProps,newState) {
    console.log('props');
    return true;
  }
  static getDerivedStateFromProps(newProps,currentState) {
    console.log('getDerivedStateFromProps');
    return null;
  }
  render() {
    console.log('render');
    return (
      <div className='container'>
        <h3>Lifecycle</h3>
        <Child />
      </div>
    )
  }
  componentDidMount() {
    console.log('componentDidMount');
  }
}

Child.jsx
import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
export default class Child extends Component {
  constructor(props) {
    super(props);
    console.log('constructor child');
  }
  shouldComponentUpdate (newProps,newState) {
    console.log('props child');
    return true;
  }
  static getDerivedStateFromProps(newProps,currentState) {
    console.log('getDerivedStateFromProps child');
    return null;
  }
  render() {
    console.log('render child');
    return (
      <div className='container'>
        <h3>Lifecycle</h3>
        <Child />
      </div>
    )
  }
  componentDidMount() {
    console.log('componentDidMount child');
  }
}
```

Console Log:

- 10 messages
- 8 user messages
- 2 errors
- No warnings
- 8 info
- No verbose

Errors:

- Unchecked runtime.lastError: The message port closed before lifecycle:44 a response was received.
- Unchecked runtime.lastError: The message port closed before lifecycle:44 a response was received.

- Trong react class component thì lifecycle mounting sẽ tự động kích hoạt theo thứ tự của lifecycle map cho dù ta có gọi - hay không gọi thì nó vẫn hoạt động.
- Điều này giúp ta có thể can thiệp vào các quá trình render trước khi giao diện xuất hiện hoặc sau khi giao diện xuất hiện. Ví dụ: API sẽ tự động load sau khi HTML của component load xong ..., hoặc handle state và props trước khi component render.

Lifecycle updating

```

import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
import Child from './Child';
export default class Lifecycle extends Component {
  constructor(props) {
    super(props);
    this.state = {
      number: 1,
      product: {
        id: 1,
        name: 'Product 1',
        price: 1000,
        img: 'https://picsum.photos/id/17/200/200'
      }
    }
    console.log('constructor');
  }

  shouldComponentUpdate(nextProps, nextState) { ... }

  static getDerivedStateFromProps(nextProps, currentState) { ... }

  render() {
    const { id, name, price, img } = this.state.product;
    console.log('render');
    return (
      <div className='container'>
        <h3>Lifecycle</h3>
        <h3>{this.state.number}</h3>
        <button onClick={()=>{
          this.setState({
            number:this.state.number + 1
          })
        }} className='btn btn-success'>increment</button>
        <Child id={id} name={name} price={price} img={img} />
      </div>
    )
  }

  componentDidMount() {
    console.log('componentDidMount');
  }
}

```

The screenshot shows a browser developer tools interface with the 'Console' tab selected. It displays the following log entries:

- getDerivedStateFromProps
- props
- render
- getDerivedStateFromProps child
- shouldComponentUpdate child
- render child

There are also status indicators: 6 messages, 6 user messages, 0 errors, 0 warnings, 6 info, and 0 verbose.

- Khi các bạn update state hoặc props sẽ dẫn đến lifecycle updating của component cha và component con kích hoạt, Vì vậy ta có thể handle việc props component thay đổi dựa vào lifecycle shouldComponentUpdate hoặc PureComponent.
- Lưu ý khi sử dụng PureComponent nhớ khi nhận được các props là reference value (object, array) thì ta phải clone object ra {...} hoặc [...] thì PureComponent mới nhận biết được.

PAGE

Lifecycle updating

Lifecycle.jsx

```
1 import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
2 import Child from './Child';
3 export default class Lifecycle extends Component {
4
5 >   constructor(props) { ... }
6
7 >   shouldComponentUpdate(nextProps, nextState) { ... }
8
9 >   static getDerivedStateFromProps(nextProps, currentState) { ... }
10
11 >   render() {
12     const { id, name, price, img } = this.state.product;
13     console.log('render');
14     return (
15       <div className='container'>
16         <h3>Lifecycle</h3>
17         <h3>{this.state.number}</h3>
18         <button onClick={()=>{
19           this.setState({
20             number:this.state.number + 1
21           })
22         }} className='btn btn-success'>increment</button>
23         {/* <Child id={id} name={name} price={price} img={img} /> */}
24         <Child product={this.state.product} />
25         <button className='btn btn-success' onClick={()=>{
26           const新产品 = {...this.state.product};
27          新产品.price = 5000;
28           this.setState({
29             product:新产品
30           })
31         }}>Set price</button>
32       </div>
33     )
34     componentDidMount() {
35       console.log('componentDidMount');
36     }
37   }
38 }
```

Child.jsx

```
1 import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
2 import { PureComponent } from 'react'; 4.2k (gzipped: 1.8k)
3
4 export default class Child extends PureComponent {
5   constructor(props) {
6     super(props);
7     this.state = {};
8     console.log('constructor child');
9   }
10
11   static getDerivedStateFromProps(nextProps,currentState) {
12     console.log('getDerivedStateFromProps child');
13     return null;
14   }
15
16   render() {
17     const {id, name, price, img} = this.props.product;
18
19     // const {id, name, price, img} = this.props;
20     // console.log('render child');
21     return (
22       <div className='container'>
23         <h3>Lifecycle child</h3>
24         <div className='card w-25'>
25           <img src={img} alt='...' />
26           <div className='card-body' >
27             <p>{name}</p>
28             <p>{price}</p>
29           </div>
30         </div>
31       </div>
32     )
33
34   componentDidMount() {
35     console.log('componentDidMount child');
36   }
37 }
```

Cập nhật giá trị phải clone ra
thì PureComponent mới hiểu
đối với object và array

Props nhận vào là object

- Cũng là ví dụ trên mình thay props khi truyền vào child là props object (this.state.product). Khi PureComponent nhận vào props là object product. Sau đó các bạn thử sử dụng onclick cập nhật price mà không clone {...} object mọi người sẽ thấy PureComponent sẽ không render lại.
 - Ngược lại nếu mọi người {...} PureComponent sẽ nhận thấy và render lại.
- ➔ Kết luận: Khi sử dụng PureComponent để tối ưu render thì cần lưu ý: nếu props là object hoặc array phải clone ra trước khi cập nhật.

Lifecycle unmount

```
reactbootcampfe > src > Pages > Lifecycle > Lifecycle.jsx > Lifecycle > timeout
You, 1 second ago | 2 authors (macbook and others)
1 import React, { Component } from 'react' 6.9k (gzipped: 2.7k)
2 import Child from './Child';
You, 1 second ago | 2 authors (macbook and others)
3 export default class Lifecycle extends Component {
4
5   >     constructor(props) { ... }
6
7   >     shouldComponentUpdate(nextProps, nextState) { ... }
8
9   >     static getDerivedStateFromProps(nextProps, currentState) { ... }
10
11   >     render() { ... }
12
13
14   >     componentDidMount() { ...
15       You, 1 second ago • Uncommitted changes
16       console.log('componentDidMount');
17       setInterval(() => {
18           console.log('fetch api');
19       }, 5000);
20   }
21
22   >     componentWillUnmount() { ...
23
24   }
25
26
27   >     timeout; You, 1 second ago • Uncommitted changes
28   >     componentDidMount() {
29       console.log('componentDidMount');
30       setInterval(() => {
31           console.log('fetch api');
32       }, 5000);
33   }
34
35   >     componentWillMount() { ...
36
37   }
38
39   >     componentWillReceiveProps(nextProps) { ...
40
41   }
42
43   >     componentWillUpdate(nextProps, nextState) { ...
44
45   }
46
47   >     componentDidUpdate(prevProps, prevState) { ...
48
49   }
50
51   >     static getSnapshotBeforeUpdate(prevProps, prevState) { ...
52
53   }
54
55   >     render() { ...
56
57   }
58
59   >     timeout; You, 1 second ago • Uncommitted changes
60   >     componentWillUnmount() { ...
61       You, 1 second ago • Uncommitted changes
62       console.log('componentWillUnmount');
63   }
64
65   >     timeout; You, 1 second ago • Uncommitted changes
66   >     componentWillUnmount() { ...
67 }
```

- Một số website có nhiều request api ngầm(websocket + api) ... Tuy nhiên khi chuyển qua component khác thì code này không tự clear.
- Trên ví dụ (fetch api) đại diện cho các đoạn code chạy ngầm.

- Component lifecycle

Cybersoft Home Shop Demo Lifecycle Form Login Sport Search Search

Lifecycle

1

increment

Lifecycle child

Product 1

1000

Set price

14 messages

14 user mess...

No errors

No warnings

14 info

No verbose

- Component form (Khi chuyển qua component khác đoạn fetch api vẫn chạy)

Cybersoft Home Shop Demo Lifecycle Form Login Sport Search Search

Home/Products/Create

Create product

Product info

id image name price description type

1 Product 1 1.000 Lorem ipsum dolor sit amet consectetur adipiscing elit. Laptop

212 messages

212 user mess...

No errors

No warnings

212 info

No verbose

Lifecycle unmount

```
* Lifecycle.jsx M X
reactbootcampfe > src > Pages > Lifecycle > Lifecycle.jsx > ...
You, 12 seconds ago | 2 authors (macbook and others)
1 import React, { Component } from 'react'  6.9k (gzipped: 2.7k)
2 import Child from './Child';
You, 12 seconds ago | 2 authors (macbook and others)
3 export default class Lifecycle extends Component {
4
5 >     constructor(props) { ...
6
7     }
8
9 >     shouldComponentUpdate(nextProps, nextState) { ...
10    }
11
12 >     static getDerivedStateFromProps(nextProps, currentState) { ...
13    }
14
15 >     render() { ...
16    }
17
18
19 >     componentDidMount() {
20         console.log('componentDidMount');
21         this.timeout = setInterval(() => {
22             console.log('fetch api');
23         }, 1000);
24     }
25
26
27 >     componentWillUnmount() {
28         //Clear Interval
29         clearInterval(this.timeout);
30     }
31
32
33 >     render() { ...
34    }
35
36
37 >     componentDidMount() {
38         console.log('componentDidMount');
39         this.timeout = setInterval(() => {
40             console.log('fetch api');
41         }, 1000);
42     }
43
44
45 >     componentWillUnmount() {
46         //Clear Interval
47         clearInterval(this.timeout);
48     }
49
50
51 >     render() { ...
52    }
53
54
55 >     componentDidMount() {
56         console.log('componentDidMount');
57         this.timeout = setInterval(() => {
58             console.log('fetch api');
59         }, 1000);
60     }
61
62
63 >     componentWillUnmount() {
64         //Clear Interval
65         clearInterval(this.timeout);
66     }
67 }
```

- Xoá các hàm chạy ngầm trước khi component mất khỏi giao diện

- Component lifecycle

Cybersoft Home Shop Demo Lifecycle Form Login Sport ▾

Search Search

Lifecycle

1

increment

Lifecycle child



Product 1

1000

Set price

Elements Recorder Console

top ▾ Filter

- ▶ ⚡ 14 messages
- ▶ 📡 14 user mess...
- ✖ No errors
- ⚠ No warnings
- ▶ 🌐 14 info
- ⚙️ No verbose

- ③ fetch api
- fetch api
- fetch api
- fetch api
- ④ fetch api
- fetch api
- fetch api
- fetch api
- fetch api

>

- Component form (Khi chuyển qua component khác đoạn code timeout đã được clear)

The screenshot shows a browser window with a navigation bar at the top. The main content area displays a 'Create product' form with fields for id, name, price, and type. Below the form is a table of products. On the right side, the developer tools' Network tab is open, showing a list of API requests and their details.

Network Tab Details:

- 14 messages
- 14 user mess...
- No errors
- No warnings
- 14 info
- No verbose
- constructor
- getDerivedStateFromProps
- render
- constructor child
- getDerivedStateFromProps child
- render child
- ComponentDidMount child
- componentDidMount
- fetch api

ID	Image	Name	Price	Description	Type
1		Product 1	1,000	Lorem ipsum dolor sit amet consectetur adipisicing elit.	Laptop

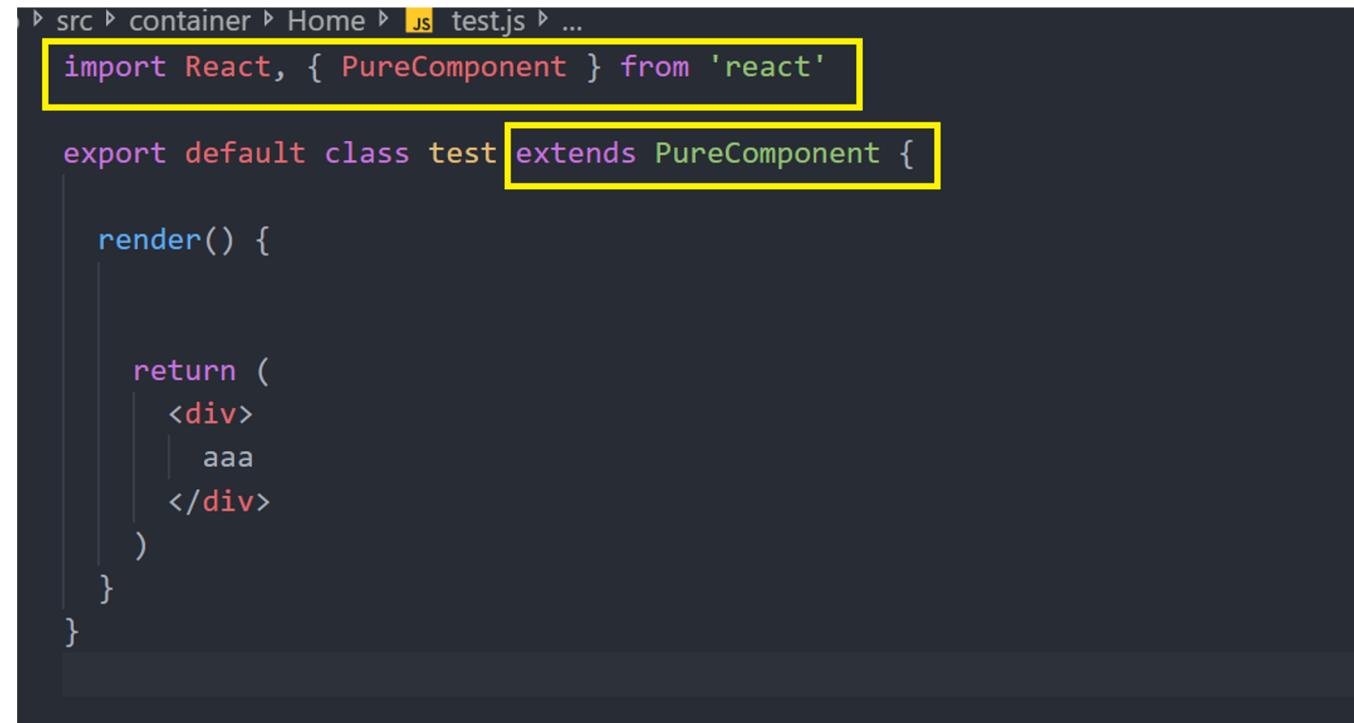
Pure component

- ❑ Trong một vài trường hợp, dù props của component không hề thay đổi nhưng vẫn bị update, dẫn tới ảnh hưởng performance của app
- ❑ Ví dụ :
- ❑ Ở đây ta có thể thấy, khi click vào nút change, state sẽ được set lại, dẫn tới hàm render() sẽ chạy lại và các component con bên trong app cũng được render lại.
- ❑ Vấn đề ở đây là state chỉ ảnh hưởng tới component DanhSachSinhVien, còn component Test vốn ko hề thay đổi, do đó việc render lại nó là ko cần thiết

```
File Edit Selection View Go Debug Terminal Help
App.js - react - Visual Studio Code
demo > src > App.js > ...
1 import React, { Component, Fragment } from 'react';
2 import { DanhSachSinhVien } from './container/danhachsinhvien';
3 import Test from './container/Home/test';
4 class App extends Component {
5   state = {
6     name: 'hieu'
7   }
8   changeName = () => {
9     this.setState({
10       name: 'Dung'
11     })
12 }
13 render() {
14   console.log('a');
15   return (
16     <div>
17       <button onClick={this.changeName}>Change</button>
18       <DanhSachSinhVien name={this.state.name} />
19       <Test />
20     </div>
21   );
22 }
23 }
24
25
26
27 export default App;
28
```

Pure component

- Để xử lý vấn đề này, ta có cách giải quyết sau
- Ở component Test, thay vì class Test extends Component, ta sẽ cho nó extends từ PureComponent, lúc này component Test chỉ render lại khi mà props của nó thật sự thay đổi



```
src > container > Home > test.js > ...
import React, { PureComponent } from 'react'

export default class test extends PureComponent {

  render() {

    return (
      <div>
        aaa
      </div>
    )
  }
}
```

➔ Lưu ý: chỉ sử dụng PureComponent, ko nên lạm dụng vì có thể dẫn tới lỗi . Bản chất của PureComponent là tự động kiểm tra xem nếu props và state của component đó thay đổi thì sẽ render lại, không thì thôi. Nhưng sự so sánh thay đổi của react là so sánh tham chiếu (shallow comparison – so sánh nguyên thủy) , nếu như ta truyền một object dưới dạng props, và thay đổi một thuộc tính nào đó thì react ko so sánh đc, vì căn bản là cùng 1 object