TRƯỜNG ĐẠI HỌC THỦY LỢI

KHOA: CÔNG NGHỆ THÔNG TIN

Ảnh có chứa Phông chữ, văn bản, biểu tượng, Nhãn hiệu

Mô tả được tạo tự động

**BÀI TẬP LỚN**

**MÔN: NỀN TẢNG PHÁT TRIỂN WEB**

**NHIỆM VỤ 01**

|  |  |
| --- | --- |
| Giảng viên: | Tạ Chí Hiếu |

Nhóm 1:

|  |
| --- |
| Vương Tiến Dũng – 2351170586 |
| Vũ Bá Đài – 2351170577 |
| Vũ Hải Đăng - 2351170580 |

Thuyết trình web (JSX & Virtual DOM) – Nhóm1

|  |  |  |  |
| --- | --- | --- | --- |
|  | TÊN THÀNH VIÊN | | |
| Vương Tiến Dũng | Vũ Bá Đài | Vũ Hải Đăng |
| NHIỆM VỤ | 1. Định nghĩa về JSX  2. Cú pháp của JSX  3. Cách biên dịch JSX sang JS thuần  4. Lợi ích và hạn chế của JSX | 1. Virtual DOM là gì ?  2. So sánh Virtual DOM với DOM thật  3. Nguyên lý hoạt động của Virtual DOM (diffing & reconciliation)  4. Lợi ích về hiệu năng của Virtual DOM | 1. Tổng kết về JSX: JSX giúp code React giống HTML nhưng cần compile  2. Tổng kết về Virtual DOM: Virtual DOM giảm thao tác trực tiếp lên DOM thật, tăng hiệu năng của re-render.  3. Demo + code |

1. **Tổng quan về JSX**
2. **Định nghĩa về JSX ?**

* JSX (JavaScript XML) là một cú pháp giúp mô tả giao diện trong React bằng cách viết mã giống HTML trong JavaScript
* JSX cho phép viết các thẻ HTML trực tiếp trong JavaScript, giúp mã nguồn dễ đọc và bảo trì hơn
* Được biên dịch thành các lệnh gọi hàm React.createElement
* Mặc dù trông giống HTML, JSX có một số nguyên tác khác biệt và yêu cầu tuân thủ cú pháp JavaScript

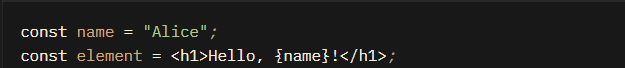
1. [**Cú pháp JSX phổ biến:**](#_top)

* các thẻ HTML phải được bao trong 1 thẻ cha (hoặc một Fragment)

A black rectangular object with white text

AI-generated content may be incorrect.

* có thể sử dụng biểu thức JavaScript trong JSX bằng cách bao trong dấu ngoặc nhọn {}



* các thuộc tính trong JSX tuân theo quy tắc camelCase (ví dụ: className, onClick)
* 
* JSX không hỗ trợ các thuộc tính chỉ định (self-closing) như HTML, cần phải có dấu gạch chéo ở cuối

A black screen with white text

AI-generated content may be incorrect.

* Ví dụ:



1. **Biên dịch JSX sang JS thuần như thế nào ?**

* Cần sử dụng công cụ **Babel –** chuyển đổi mã phổ biến trong hệ sinh thái Javascript
* **Babel** giúp chuyển đổi JSX thành các lệnh gọi hàm React.createElement mà trình duyệt có thể hiểu
* Bên cạnh đó, **Babel** còn được tích hợp cùng Webpack hoặc các trình biên dịch hiện đại khác trong môi trường phát triển React
* **Webpack**: Được sử dụng để đóng gói và tích hợp Babel với các tập tin trong dự án
* Ví dụ khi một đoạn mã JSX được biên dịch qua Babel:

const element = <h1>Hello, world!</h1>;

* const element = React.createElement('h1', null, 'Hello, world!');

Trong đó:

“h1” là phần tử cần tạo

“null” là props được truyền vào (ở đây trống)

“Hello, world!” là nội dung con

* Ví dụ khi một đoạn mã JSX được biên dịch qua Babel, có thêm props có giá trị

const element = <h1 name = “65KTPM”>Hello, world!</h1>;

* const element = React.createElement('h1', {name: 65KTPM}, 'Hello, world!');

**Sơ lược về React**

* Là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng (UI) cho các ứng dụng mobile hoặc web
* Cho phép xây dựng các thành phần (components) độc lập, có thể tái sử dụng và dễ dàng bảo trì
* Tiếp cận “hướng thành phần” và sử dụng Virtual DOM để tăng hiệu năng khi cập nhật giao diện

1. **Lợi ích và hạn chế của JSX là gì ?** 
   1. **Lợi ích của JSX**

* **Dễ đọc và trực quan**: Cú pháp giống HTML giúp mô tả giao diện người dùng một cách rõ ràng, dễ hiệu, dễ tiếp cận với người đã đã biết HTML
* **Tích hợp JavaScript**: Giao diện được mô tả trực tiếp trong mã JavaScript (dùng {}), không cần tách file HTML riêng, kết hợp logic và dữ liệu động dễ dàng
* **Hỗ trợ component**: Tiện lợi khi xây dựng và tái sử dụng các thành phần giao diện trong react
* **Kiểm tra lỗi compile sớm**: Trình biên dịch (như Babel) sẽ báo lỗi sớm khi cú pháp JSX sai, tránh lỗi runtime.
* **Tối ưu hóa hiệu suất**: Khi kết hợp với React, JSX được biên dịch thành các cấu trúc hiệu quả, giúp cải thiện tốc độ render
* **Dễ bảo trì và tái sử dụng code**: Giao diện và logic liên quan nằm cùng một nơi, dễ quản lý theo component
  1. **Hạn chế của JSX**
* **Phụ thuộc vào công cụ biên dịch**: Cần công cụ như Babel để chuyển đổi JSX thành JavaScript, làm phức tạp quá trình thiết lập dự án
* **Khác biệt với HTML**: Yêu cầu học các quy tắc riêng (như className thay vì class), có thể gây nhầm lẫn cho người mới
* **Kích thước mã**: Sử dụng JSX có thể làm mã nguồn dài hơn so với cách viết thuần JavaScript (như React.createElement)
* **Hạn chế tính linh hoạt**: Không phải mọi logic phức tạp đều dễ thực hiện trực tiếp trong JSX, đôi khi cần tách ra ngoài
* **Khó đọc khi logic phức tạp**: Nếu lồng nhiều biểu thức hoặc điều kiện trong JSX, code dễ bị rối và khó bảo trì.

1. **Virtual DOM**
2. **Virtual DOM là gì ?**
   1. **DOM là gì ?**



* **DOM** (**D**ocument **O**bject **M**odel) là một biểu diễn của cấu trúc của một trang web HTML hoặc XML. Nó đại diện cho các phần tử trên trang web như các thẻ HTML, các thuộc tính và văn bản trong trình duyệt. Khi bạn thay đổi DOM bằng JavaScript, trình duyệt cần phải làm lại việc vẽ (re-render) các phần tử trên trang, điều này có thể ảnh hưởng đến hiệu suất ứng dụng của bạn.
* **DOM** giúp thao tác với dữ liệu theo mô hình hướng đối tượng do các phần tử trong **DOM** có cấu trúc được định nghĩa thành các *đối tượng, phương thức, thuộc tính* để có thể truy xuất dễ dàng. Chúng được coi như các *node* và được biểu diễn dưới dạng **DOM Tree**.
  1. **Virtual DOM là gì ?**
* **Virtual DOM** (**V-DOM**) là một bản sao ảo của DOM thật, được giữ trong bộ nhớ dưới dạng JavaScript objects.
* Nó là một biểu diễn JavaScript của cấu trúc DOM và sử dụng để theo dõi sự thay đổi trong ứng dụng React. Khi bạn thay đổi trạng thái của ứng dụng, React tạo ra một **Virtual DOM** mới.

1. **So sánh Virtual DOM và DOM thật**

* **DOM** là viết tắt của mô hình đối tượng tài liệu , đây là cấu trúc mà trình duyệt sử dụng để biểu diễn tất cả thành phần trên trang web như tiêu đề , đoạn văn và nút vì vậy khi bạn thay đổi cái gì đó trên trang , trình duyệt sẽ cập nhật **DOM** trực tiếp điều này có thể chậm vì nó có thể vẽ lại nhiều thành phần trong React ,
* Trong **Virtual DOM** khi thay đổi xảy ra như nhấp vào nút nhưng để cập nhật tiêu đề , React trước tiên sẽ cập nhật **DOM** ảo , sau đó kiểm tra xem có gì khác so với phiên bản trước đó không , quá trình này được gọi **diffing** (khác biệt) , cuối cùng React chi cập nhật phầm DOM thực đã thay đổi giúp cập nhật nhanh hơn
* **Ví dụ :** Khi bạn thay đổi tiêu đề, React chỉ cập nhật tiêu đề đó thay vì tải lại mọi thứ giúp người dùng có trải nghiệm mượt mà hơn .

1. **Nguyên lý hoạt động của Virtual DOM (diffing & reconciliation)**

* Khi có sự thay đổi :
  1. **Khởi tạo** : React sẽ tạo một Virtual DOM hoàn chỉnh mới dựa trên dữ liệu mới
  2. **Render component** : khi trạng thái của người dung thay đổi ; ví dụ người dùng ấn nút , React sẽ tạo ra một cây Virtual DOM mới cho trang web / ứng dụng
  3. **So sánh** : React so sánh cây Virtual DOM mới so với Virtual DOM cũ bằng cách sử dụng thuật toán **deffing** (deffing là quá trình tìm điểm khác biệt giữa 2 cây và xác định những thay đổi cần thực hiện trên DOM thực tế để cập nhật nó ).
  4. **Cập nhật DOM ( reconciliation )** : sau khi xác nhận được sự khác biệt , React chỉ cập nhật phần tử DOM thực tế mà cần thay đổi thay vì cập nhật toàn bộ trang web , điều này giúp tối ưu hóa hiệu suất và làm giảm thời gian không cần thiết để vẽ lại trang web.

1. **Lợi ích về hiệu năng của Virtual DOM**

* **Tối ưu hiệu suất :** nhanh hơn DOM thật , giảm số lần truy cập và thao tác trực tiếp với DOM thật , giúp trang web , ứng dụng chạy nhanh và mượt hơn
* **Tránh reflow và repaint không cần thiết:** chỉ cập nhật phần giao diện thực sự thay đổi
* **Lập trình dễ dàng :** tự động cập nhật UI mà không cần thao tác thủ công trên DOM , code rõ ràng
* **Virtual DOM** giúp **R**eact và các framework tương tự tăng tốc độ hiển thị giao diện , tối ưu hóa hiệu suất và làm cho việc phát triển ứng dụng dễ dàng hơn

1. **Tổng kết về JSX & Virtual DOM**
2. **JSX là gì ?**

* JSX (JavaScript XML) là cú pháp mở rộng của JavaScript
* JSX chod phép viết các đoạn mã giống HTML ngay trong JavaScript, giúp mô tả UI dễ dàng và trực quan hơn

1. **Lợi ích của việc sử dụng JSX**

* Trực quan và quen thuộc: Lập trình viên có thể viết giao diện giống HTML nên dễ đọc, dễ hiểu, đặc biệt với người quen HTML/CSS
* Gắn liền với logic: JSX cho phép nhúng các biểu thức JavaScript bên trong {} nên dễ dàng kết hợp giữa giao diện và logic xử lý.
* Cộng đồng và công cụ hỗ trợ tốt: JSX được cộng đồng React hỗ trợ mạnh mẽ, dễ tích hợp với các công cụ như Babel và Webpack.

1. **JSX cần được biên dịch**

* Trình duyệt không hiểu JSX trực tiếp, do đó JSX phải được biên dịch thành JavaScript thuần (thường là lệnh gọi React.createElement())
* Việc biên dịch thường được thực hiện bởi công cụ như Babel trong quá trình build ứng dụng React
* Ví dụ về biên dịch sử dụng công cụ Babel:



Được biên dịch thành:

A screen shot of a computer program

AI-generated content may be incorrect.

1. **Tổng kết**

* JSX giúp code React giống HTML, tăng tính trực quan và dễ bảo trì
* Tuy nhiên, nó không phải là HTML thật, và cần được biên dịch về JavaScript trước khi trình duyệt hiểu được

1. **Virtual DOM là gì ?**

* Là một bản sao ảo (virtual) của DOM thật (real DOM)
* Là một cấu trúc dữ liệu dạng cây đại diện cho giao diện UI hiện tại của ứng dụng

1. **Vấn đề với DOM thật**

* DOM thật chậm trong việc cập nhật khi có nhiều thay đổi liên tục
* Việc thao tác trực tiếp lên DOM (thêm, xóa, cập nhật node) gây tốn tài nguyên và làm giảm hiệu năng

1. **Cách Virtual DOM cải thiện hiệu năng**

* Khi UI thay đổi, React không cập nhật trực tiếp DOM thật ngay lập tức
* Thay vào đó:

1. React tạo ra một Virtual DOM mới phản ánh giao diện sau thay đổi
2. So sánh (diffing) giữa Virtual DOM cũ và mới
3. Chỉ áp dụng những thay đổi tối thiểu cần thiết lên DOM thật (reconciliation)

|  |  |
| --- | --- |
| **Giai đoạn** | **Mô tả nội dung** |
| **Trạng thái trước** | Virtual DOM 1: <ul><li>A</li><li>B</li></ul> |
| **Cập nhật State** | Virtual DOM 2: <ul><li>A</li><li>C</li></ul> |
|  | Diffing: So sánh DOM cũ và mới |
| **Cập nhật DOM thật** | Real DOM: Chỉ cập nhật <li>B</li>🡪<li>C</li> |

1. **Lợi ích của Virtual DOM**

* Giảm số lần thao tác trực tiếp lên DOM thật
* Tối ưu hiệu năng khi render lại UI
* Cải thiện trải nghiệm người dùng, nhất là trong các ứng dụng lớn hoặc có UI phức tạp

1. **Tổng kết**

* Virtual DOM là một cách tiếp cận thông minh để giảm tải cho DOM thật
* Nhờ vào cơ chế diff & reconciliation, React chỉ cập nhật phần giao diện cần thiết, giúp ứng dụng chạy mượt hơn và nhanh hơn