**1. ES6 là gì?**

ES6 (ECMAScript 2015) là phiên bản thứ 6 của JavaScript, giới thiệu nhiều tính năng mới và cải tiến giúp việc viết mã JavaScript dễ dàng, ngắn gọn và hiệu quả hơn.

**2. Tính năng mới của ES6?**

* **let và const**: Khai báo biến với phạm vi block scope.
* **Arrow function**: Viết hàm ngắn gọn và không có binding của this.
* **Template literals**: Chuỗi template sử dụng dấu backtick (``) cho phép chèn biến trực tiếp.
* **Destructuring**: Trích xuất dữ liệu từ object/array.
* **Default parameters**: Giá trị mặc định cho tham số hàm.
* **Spread/rest operators**: Giải/phân tách mảng hoặc object.
* **Classes**: Định nghĩa class theo cách dễ hiểu, hỗ trợ kế thừa.
* **Modules (import/export)**: Chia code thành nhiều file và quản lý dễ dàng.

**3. Phân biệt map và forEach**

* **forEach**: Lặp qua từng phần tử nhưng không trả về mảng mới.
* **map**: Lặp và trả về một mảng mới từ các phần tử được biến đổi.  
  **Khác biệt chính**: map có return, forEach thì không.

**4. Reducer, filter, find**

* **reduce()**: Gom tất cả các phần tử thành một giá trị duy nhất (sum, average, etc).
* **filter()**: Lọc các phần tử thỏa mãn điều kiện và trả về mảng mới.
* **find()**: Tìm phần tử đầu tiên thỏa mãn điều kiện và trả về phần tử đó.

**5. React là gì? Ưu điểm và nhược điểm?**

**React** là thư viện JavaScript dùng để xây dựng giao diện người dùng (UI).

* **Ưu điểm**: Tái sử dụng component, DOM ảo tăng hiệu năng, cộng đồng lớn.
* **Nhược điểm**: Học ban đầu khó, cần cấu hình build, cập nhật nhanh.

**6. SEO là gì?**

SEO (Search Engine Optimization) là tối ưu hóa trang web để đạt thứ hạng cao trên công cụ tìm kiếm như Google.

**7. SSR, CSR? Ưu nhược điểm?**

* **SSR (Server Side Rendering)**: Render trên server, tải nhanh, SEO tốt.
* **CSR (Client Side Rendering)**: Render trên client, tải chậm lúc đầu nhưng nhanh trong tương tác.  
  **Ưu điểm SSR**: SEO tốt, tải trang nhanh.  
  **Ưu điểm CSR**: Tương tác mượt mà sau khi tải lần đầu.

**8. DOM ảo?**

DOM ảo là bản sao của DOM thật, giúp React cập nhật UI hiệu quả mà không cần render lại toàn bộ trang.

**9. JSX là gì? Vì sao nên dùng JSX?**

JSX (JavaScript XML) cho phép viết HTML trong JavaScript, giúp code dễ đọc và dễ bảo trì.

**10. Trình duyệt có đọc được JSX hay không?**

**Không.** JSX cần được biên dịch thành JavaScript bằng Babel trước khi trình duyệt có thể hiểu.

**11. Thư viện Babel**

Babel là trình biên dịch mã, giúp dịch JSX thành JavaScript thuần.

**12. Không có JSX có code React được không?**

**Được.** Dùng React.createElement() để tạo component mà không cần JSX.

**13. Lưu ý khi sử dụng JSX**

* class → className.
* Sự kiện viết dưới dạng camelCase.
* JSX phải return về một thẻ duy nhất.

**14. Làm sao có thể viết mã JS trong JSX**

Viết code JavaScript trong JSX bằng cặp {}:

jsx

Copy code

const name = 'John';

<h1>{name}</h1>

**15. Component là gì?**

Component là thành phần UI độc lập, có thể tái sử dụng.

**16. Các loại component?**

* **Function Component**
* **Class Component**

**17. Phân biệt function và class Component?**

* **Function Component**: Nhẹ, dễ viết, dùng hook.
* **Class Component**: Có state, lifecycle method nhưng phức tạp hơn.

**18. State? Cách cập nhật state?**

State là dữ liệu bên trong component. Cập nhật bằng setState hoặc useState trong function component.

**19. State thay đổi thì điều gì xảy ra?**

Component được render lại.

**20. Props là gì?**

Props là dữ liệu truyền từ component cha sang component con.

**21. Phân biệt state và props?**

* **State**: Nội bộ, có thể thay đổi.
* **Props**: Bên ngoài, không thể thay đổi trong component con.

**22. Lifecycle method?**

Phương thức quản lý vòng đời component (mount, update, unmount).

**23. Hook là gì? Mục đích ra đời?**

Hook cho phép dùng state và lifecycle trong function component mà không cần class.

**24. Một số hook đã học**

* useState, useEffect, useRef, useParams, useNavigate, useLocation.

**25. useState()? Cách cập nhật state?**

Khai báo và cập nhật state trong function component:

jsx

Copy code

const [count, setCount] = useState(0);

setCount(count + 1);

**26. useEffect? Các dependency?**

useEffect chạy khi component render hoặc cập nhật. Dependency quyết định khi nào useEffect chạy lại.

**27. Router dùng để làm gì? Cách triển khai?**

Router điều hướng trong React. Dùng react-router-dom để tạo route.

**28. formik và yup dùng để làm gì?**

* **Formik**: Quản lý form.
* **Yup**: Validation form.

**29. Bất đồng bộ là gì? Cách xử lý?**

Bất đồng bộ là khi code không chạy tuần tự. Xử lý bằng:

* Callback
* Promise
* Async/await

**30. axios là gì?**

Axios là thư viện HTTP client giúp gọi API dễ dàng.

**31. Redux là gì?**

Redux là thư viện quản lý state tập trung.

**32. Khi nào sử dụng redux?**

Khi ứng dụng lớn, state phức tạp và cần chia sẻ giữa nhiều component.

**33. Các thành phần của 1 redux**

* Store, Reducer, Action.

**34. Middleware là gì? Khi nào dùng?**

Middleware xử lý logic trung gian khi dispatch action (ví dụ: redux-thunk).

**35. SSR, CSR, SSG**

* SSR: Render trên server.
* CSR: Render trên client.
* SSG: Static site generation (trang tĩnh trước khi deploy).

**36. Tính năng của Next.js**

* SSR, SSG, API route, hỗ trợ TypeScript, tối ưu SEO.