# JavaScript Advanced

1. hoisting -> ok
2. arrow function -> ok
3. IIFE -> ok
4. strict mode -> ok
5. toán tử spread -> ok
6. toán rest -> ok
7. Destructuring -> ok
8. truthy and falsy -> ok
9. closure -> ok
10. callback -> ok
11. prototype -> ok
12. polyfill -> ok
13. module -> ok
14. this -> ok
15. Promise (Sync/Async) -> ok
16. fetch -> ok
17. target -> ok

# HTML DOM, React DOM

1. document.createElement()
   1. document.createElement()
   2. document.body.appendChild()
   3. console.dir()
   4. innerText, innerHTML, id, className, style (Object.Assign())
2. Cách thêm React vào website
   1. Github, NPMJS, UNPKG
   2. Add react to website (Đối tượng React)
   3. Official documents (reactjs.org)
3. React.createElement()
   1. React.createElement(type, props, …childrens) -> React Element
   2. So sánh với document.createElement -> DOM element
   3. Thay đổi Id, className, Style,…
   4. Thêm Text, HTML
   5. Ví dụ:
      1. <h1 title=”Hello” class=”heading”>Hello guys</h1>
      2. <ul><li>JavaScript</li></ul>
      3. <div class=”post-item”>

<h2 title=”React”>React</h2>

<p>CodeGym</p>

<div>

1. ReactDOM
   1. Tại sao phải sử dụng React-DOM
   2. Tại sao tách React-DOM ra
   3. Render UI
      1. ReactDOM.render(reactElement , root)
2. JSX
   1. Luôn dùng React.createElement() -> vấn đề
   2. JSX không phải là HTML?
      1. Cần có JavaScript, Babel để dùng JSX (https://babeljs.io/repl)
   3. JSX và React-DOM
3. React element types: string, function/class -> tạo ra component
   1. Wrapper
      1. Header component
      2. Content component
      3. Footer component
   2. Biểu diễn đơn giản với JSX (sử dụng function component và class component)
4. Props
   1. React Element
      1. Sử dụng props giống như với attribute của thẻ HTML
      2. 2 props class, for => className, htmlFor
      3. Phải tuân thủ theo quy ước có sẵn
   2. React components
      1. Sử dụng giống như đối số cho Component
      2. Tự do đặt tên props
         1. Đặt theo camelCase
         2. Có thể bao gồm dấu gạch ngang
   3. Chú ý
      1. Props “key” là prop đặc biệt
      2. Prop cơ bản là đối số của Component
         1. Prop có thể là bất cứ kiểu dữ liệu gì
      3. Sử dụng destructuring
5. …

# Hooks

1. useState(): thay đổi trạng thái dữ liệu

+ Component được re-render sau khi setState

+ Initial state chỉ dùng cho lần đầu

+ Set state với callback

+ Initial state với callback

+ Set state là thay thế state bằng giá trị mới

--------------------------------------

+ Ví dụ 1: random gift

+ Ví dụ 2: todolist

+ Ví dụ 3: to-waybinding (Input, Radio, Checkbox)

1. useEffect(): thay đổi/ ảnh hưởng tình trạng trang web

* Cập nhật lại state
* Cập nhật DOM (mounted)
* Render lại UI
* Gọi cleanup nếu deps thay đổi
* Gọi useEffect callback
* callback luôn được gọi sau khi component mounted
* cleanup function luôn được gọi trước khi component unmounted
* cleanup function luôn được ọi trước khi callback được gọi (trừ lần mounted)
  1. useEffect(callback)
     1. Gọi callback mỗi khi component re-render
     2. Gọi callback sau khi component thêm element vào DOM
  2. useEffect(callback, [])
     1. Chỉ gọi callback 1 lần sau khi component mounted
  3. useEffect(callback, [deps])
     1. Callback sẽ được gọi lại mỗi khi dependency thay đổi
  4. …

Ví dụ áp dụng:

* + 1. Update DOM: Thay đổi title của web
    2. Call API
    3. Listen DOM events: Scroll, Resize
    4. Cleanup: remove listener/unsubscribe; clear timers

1. useLayoutEffect()
   1. Cập nhật lại state
   2. Cập nhật DOM (mutated)
   3. Gọi cleanup nếu deps thay đổi (sync)
   4. Gọi useLayoutEffect callback (sync)
   5. Render lại UI
2. useRef()
   1. Lưu các giá trị qua 1 tham chiếu bên ngoài
   2. Function component