06. React Part1

Attention!

This training is **interview-driven**.

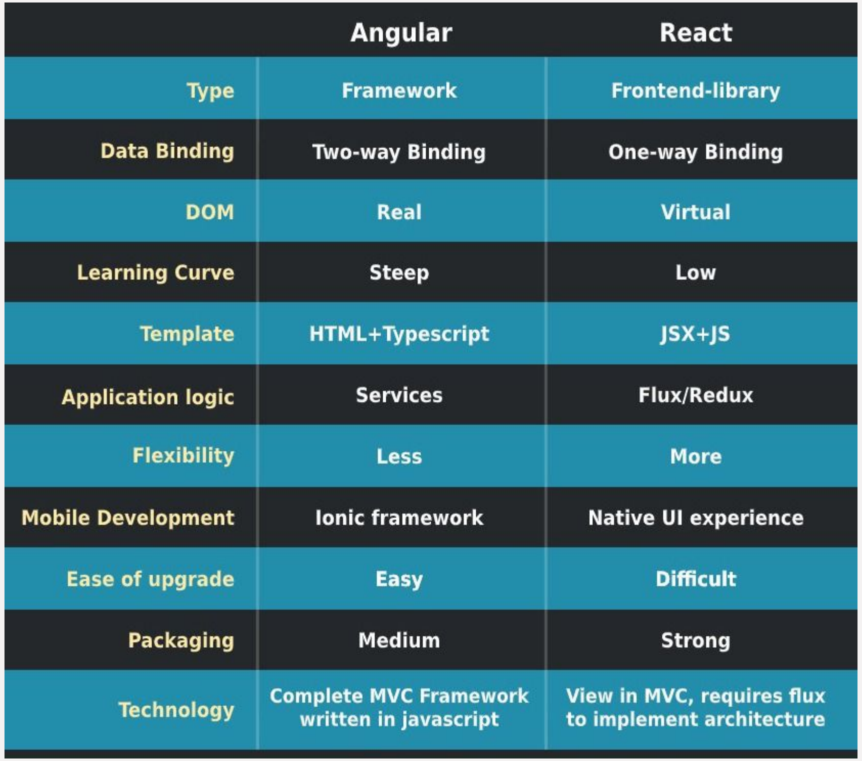
To become a qualified developer, *ChatGPT* and *YouTube* will always be your best friends.

Outline

- React vs Angular
- JSX
- React Fragment
- Component
- Props, State
- Virtual DOM, Shadow DOM
- SPA
- Lifecycle
- Pure Component
- CSR vs SSR

React vs Angular

- **React:** an open-source JavaScript **library** developed by Facebook/Meta for building user interfaces
    - library, Virtual DOM, JSX, SPA, One-way Data Binding, Hooks
- **Angular:** a front-end **framework** developed by Google for building dynamic web applications
    - framework, model/view, Two-Way Data Binding, DI, Directives, Routing



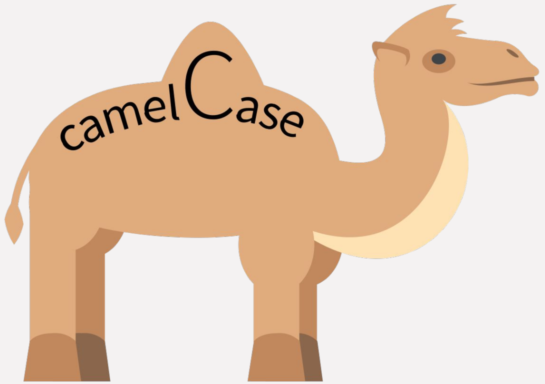
	Angular	React
Type	Framework	Frontend-library
Data Binding	Two-way Binding	One-way Binding
DOM	Real	Virtual
Learning Curve	Steep	Low
Template	HTML+Typescript	JSX+JS
Application logic	Services	Flux/Redux
Flexibility	Less	More
Mobile Development	Ionic framework	Native UI experience
Ease of upgrade	Easy	Difficult
Packaging	Medium	Strong
Technology	Complete MVC Framework written in javascript	View in MVC, requires flux to implement architecture

React Basic Concepts

Component	Reusable UI unit	`<Header />`, `<Button />`
JSX	JavaScript + XML syntax	`<div>Hello</div>`
Props	Input parameters passed to a component	`<Hello name="Tom" />`
State	Internal data/state of a component	`useState` to manage data
Hook	A way to use state and logic in function components	`useState`, `useEffect`, etc.
Virtual DOM	Improves update performance	Automatic diff & update

JSX

- JSX (JavaScript XML): A syntax allows to write HTML-like syntax directly in JS
- Compilation: transpile it into React.createElement
- **Rules**:
  - Return One Parent Element
    - <div></div>
    - <React.Fragment></React.Fragment>
    - <></>
  - Use {} to embed JavaScript expressions
  - Attributes Use CamelCase

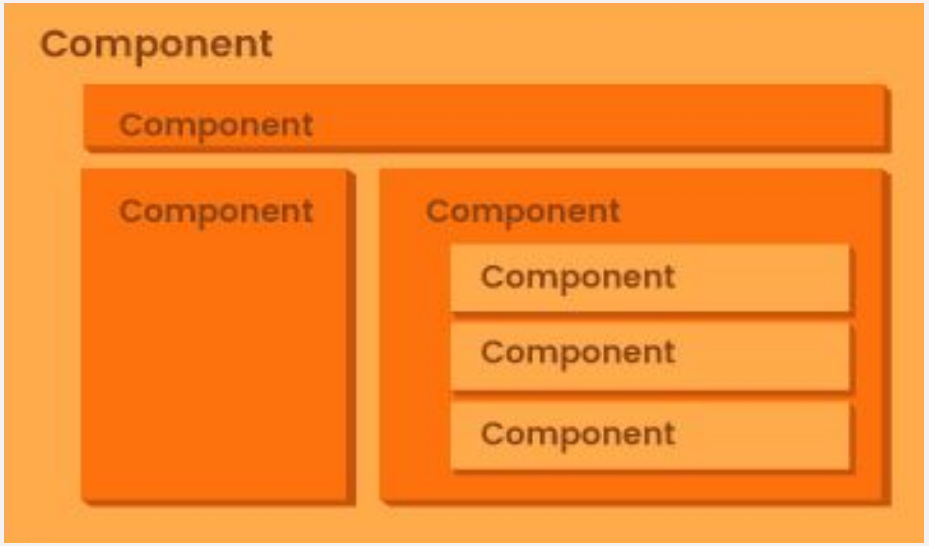


React Fragment

- A React Fragment is a special type of wrapper that lets you **group multiple elements** without adding an extra DOM node
- A component must return only one parent element
  - <div> wrapper
    - Adds an extra <div> in the DOM
  - React Fragment
    - Groups the elements without rendering an actual wrapper in the DOM
    - Short Syntax: <></>
      - Short syntax does **NOT** support keys or attributes

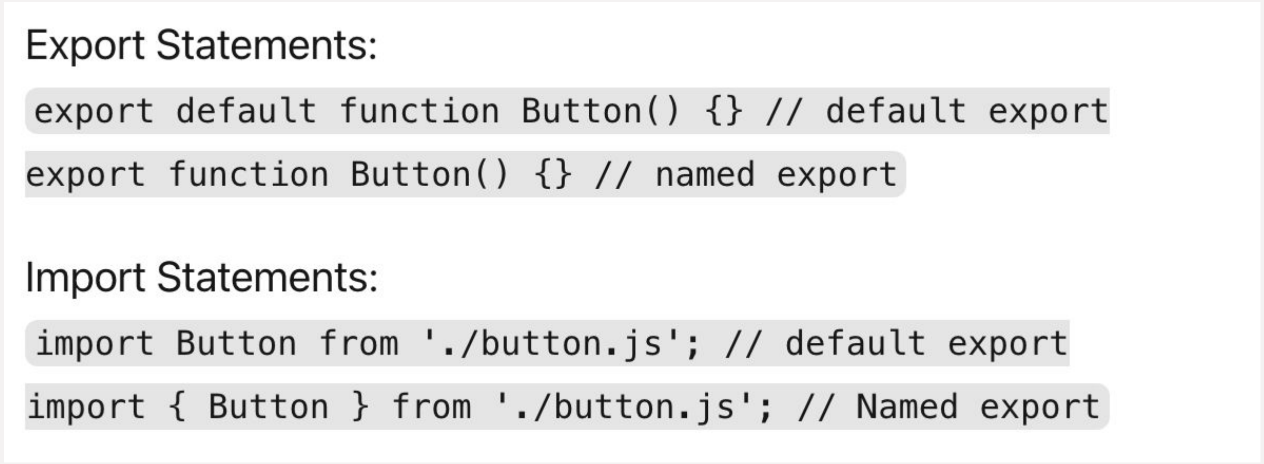
Component

- Component: a section of the user interface
  - Each component has its own logic, structure, and style
  - Independent, reusable (**DRY**)
- 2 main types
  - **Functional** Components
  - Class Components
- Benefits
  - Scalability, Maintainability, Testability, Collaboration



Component: Import/Export

- **Default**
  - **NO** curly braces {} are used
  - Can rename the import to anything
- **Named**
  - Can Export multiple values from a file
  - Must **USE** curly braces {}
  - Must match the exported names exactly



Export Statements:

```
export default function Button() {} // default export
export function Button() {} // named export
```

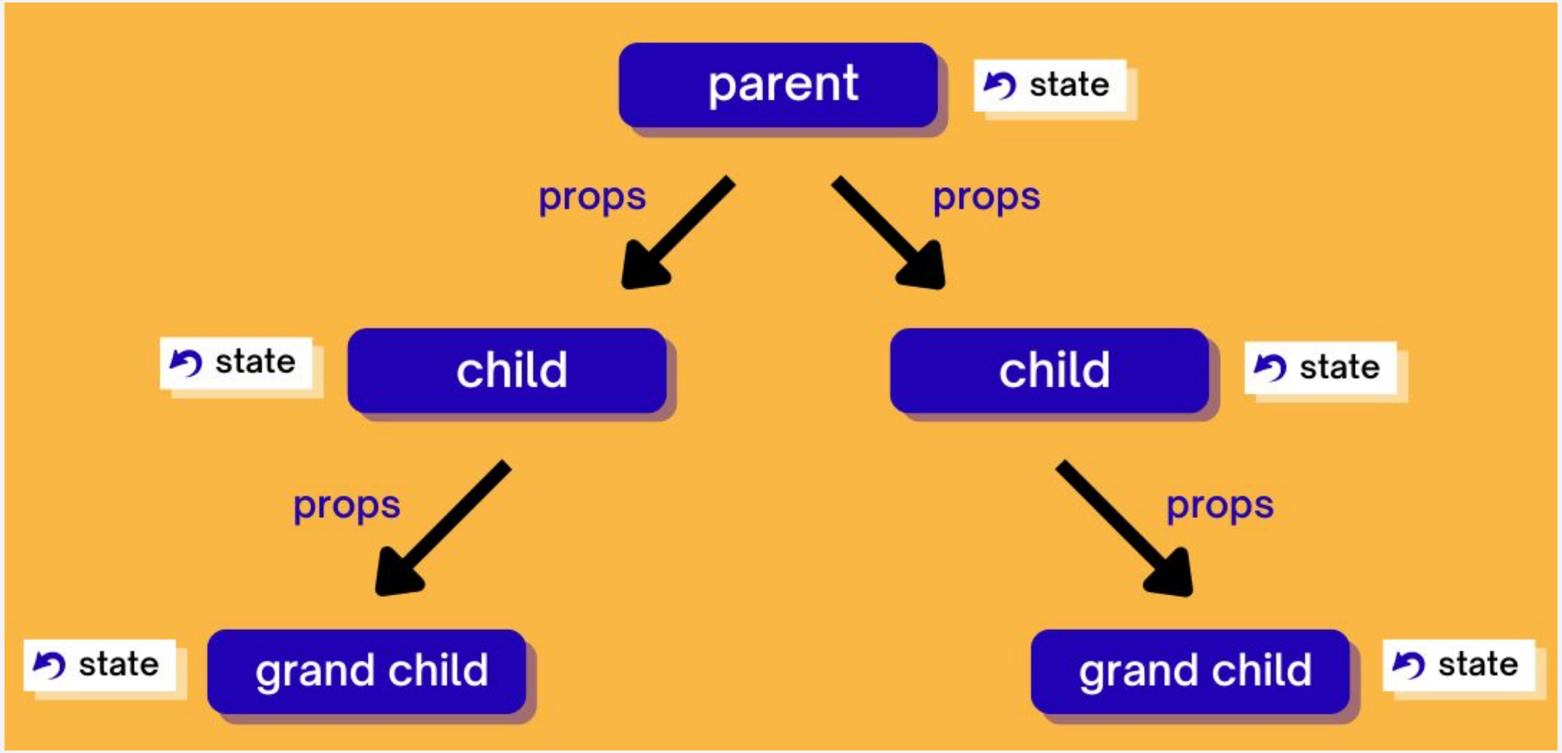
Import Statements:

```
import Button from './button.js'; // default export
import { Button } from './button.js'; // Named export
```

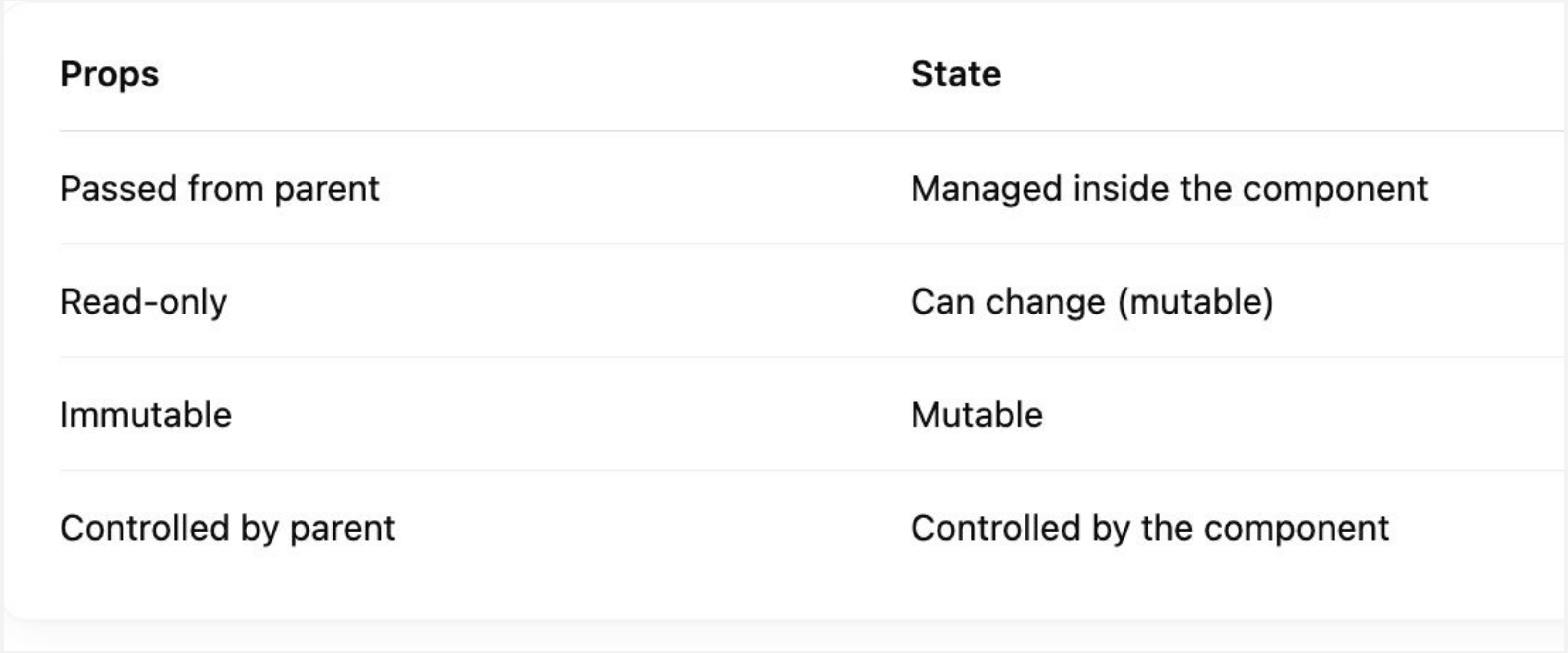
Props vs State

- **Props** (properties)
    - Inputs passed from a parent component to a child component
    - Props are **read-only**
        - Cannot modify props in **child** component
    - **Unidirectional Data Binding**: Props are passed from parent to child only
    - **Destructuring** Props
- **State**: Data owned and managed by a component itself
        - **Mutable**: State is mutable and causes re-rendering when updated
        - Hooks (useState)/this.setState are used to manage state
        - Changing state triggers UI re-render

Props vs State



Props vs State



Props	State
Passed from parent	Managed inside the component
Read-only	Can change (mutable)
Immutable	Mutable
Controlled by parent	Controlled by the component

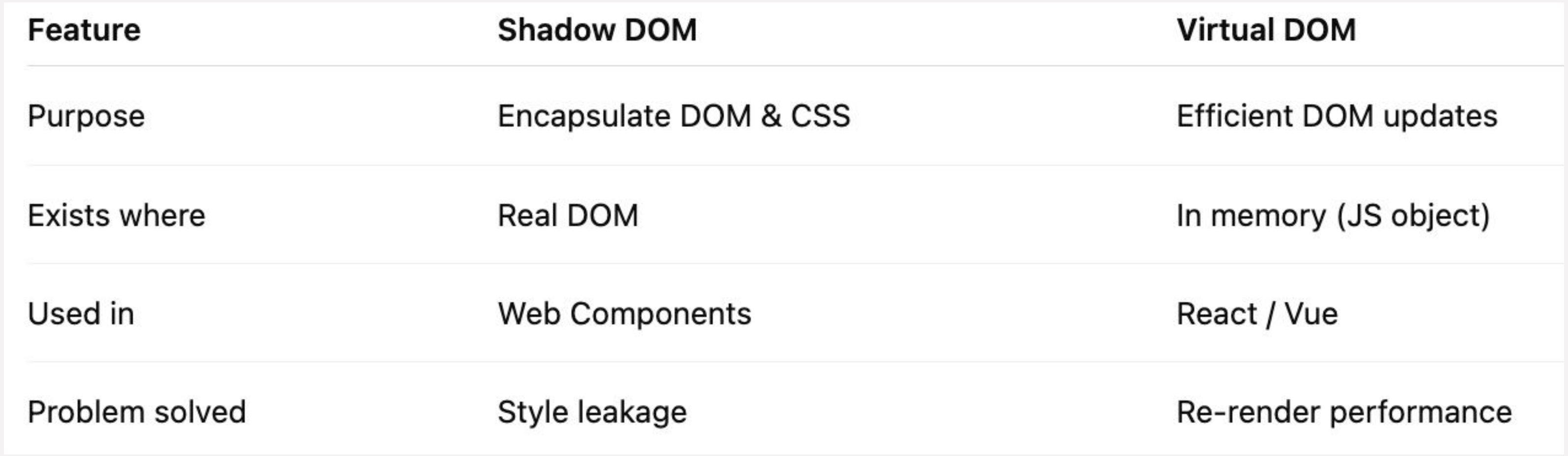
Virtual DOM

- Virtual DOM (VDOM): a lightweight copy of the real DOM
- ❌ Updating the browser DOM directly
- Diffing algorithm
  - Creates a Virtual DOM
  - Compares new and old Virtual DOM trees
    - Same types are reused
    - different types are replaced
    - **keys** help correctly match list items to minimize DOM updates

```
State / Props change
          ↓
Render → New Virtual DOM
          ↓
Diff old vs new Virtual DOM
          ↓
Compute minimal changes
          ↓
Commit updates to real DOM
```

Virtual DOM vs Shadow Dom

- Shadow DOM is a small “box” for DOM and CSS
    - Styles and structure are isolated
    - Outside code can’t touch it, and it doesn’t affect the outside
    - Prevents CSS conflicts

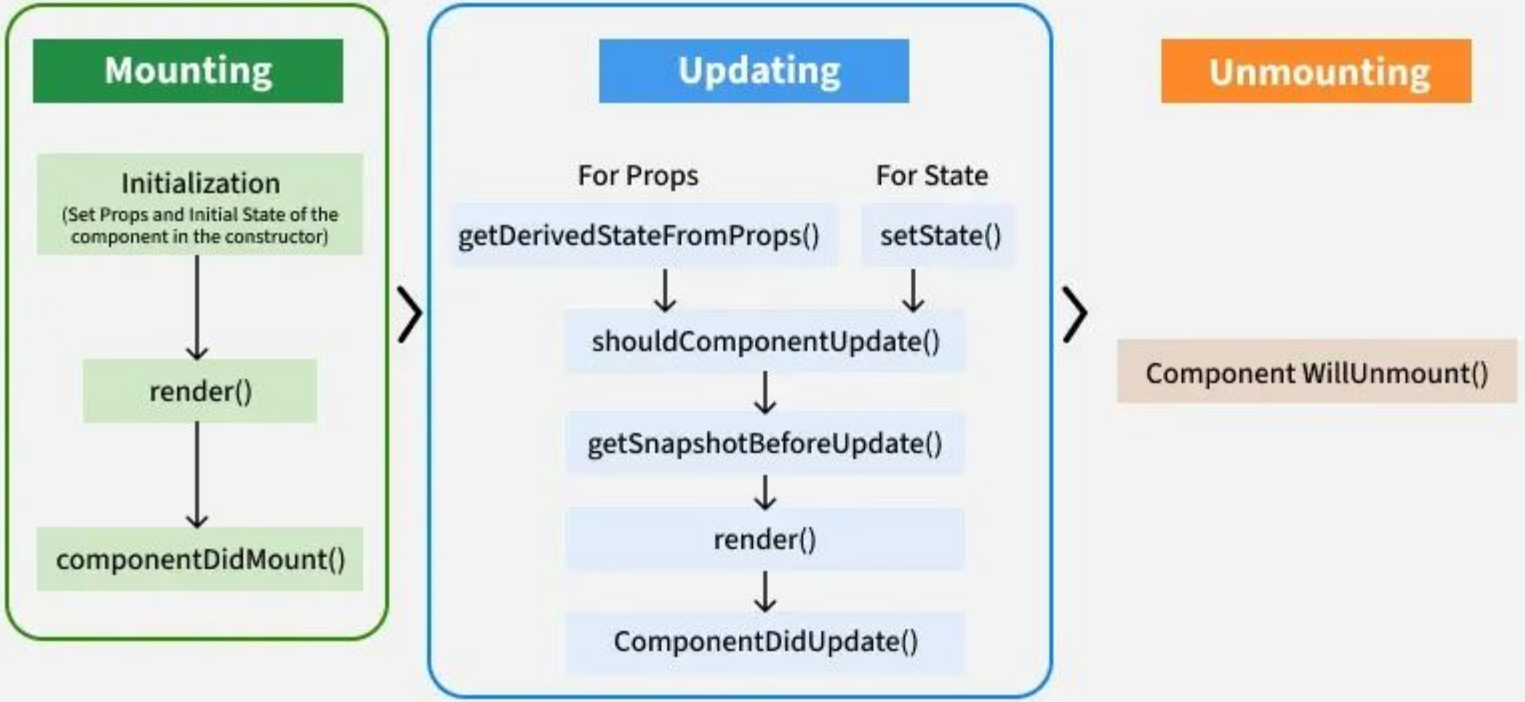


Feature	**Shadow DOM**	**Virtual DOM**
Purpose	Encapsulate DOM & CSS	Efficient DOM updates
Exists where	Real DOM	In memory (JS object)
Used in	Web Components	React / Vue
Problem solved	Style leakage	Re-render performance

React Lifecycle

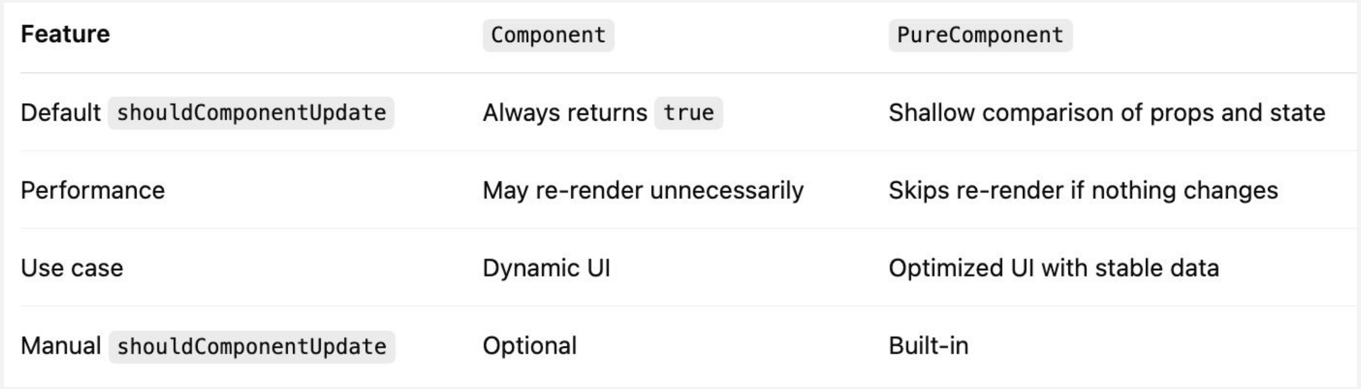
- The stages that a component follows from when it appears on the page to when it is removed
- Lifecycle Phases
    - **Mounting** (creation and inserted into DOM)
        - When the component is created and added to the DOM
    - **Updating** (component re-renders due to props/state change)
    - When the component is being re-rendered due to changes in props/state
    - **Unmounting** (component removed from DOM, cleanup)
        - When the component is being removed from the DOM
- Class components use **lifecycle methods**, while functional components use **useEffect** to handle side effects

React Lifecycle Methods (Class Component)



Pure Component

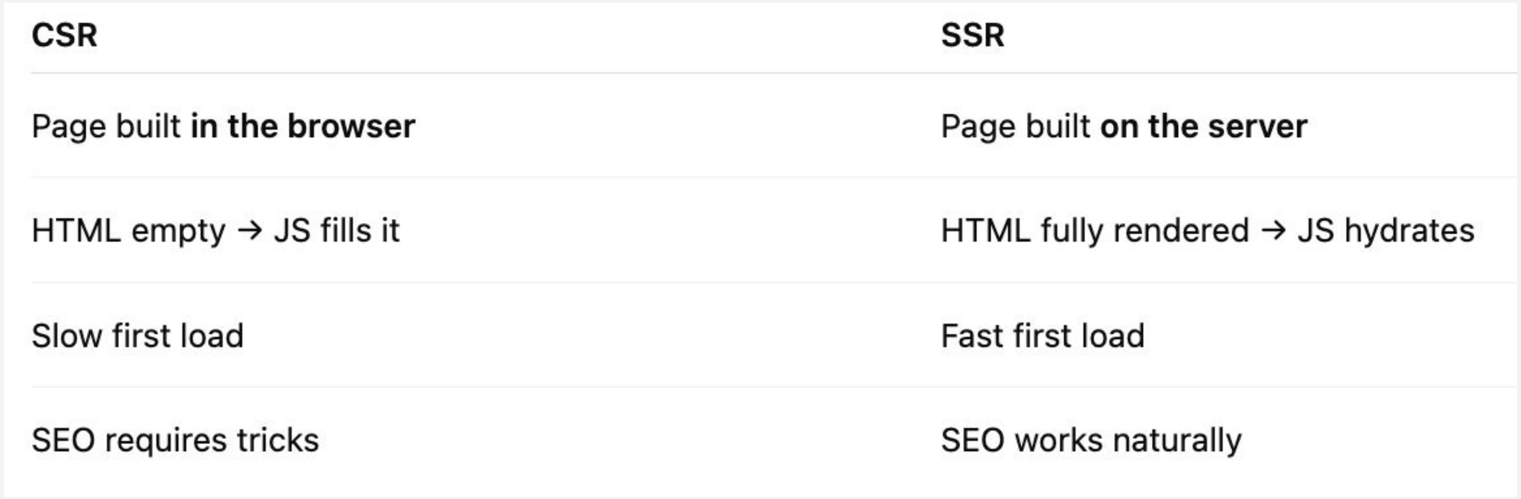
- A Pure Component in React is a **class component** that automatically implements a **shallow comparison** in **shouldComponentUpdate**
    - **Syntax:** class MyComponent extends React.PureComponent {}
    - Only re-renders when its **props** or **state** change (If props and state don’t change, the component won’t re-render)
    - **Performance optimization** by avoiding unnecessary re-renders



Feature	`Component`	`PureComponent`
Default `shouldComponentUpdate`	Always returns `true`	Shallow comparison of props and state
Performance	May re-render unnecessarily	Skips re-render if nothing changes
Use case	Dynamic UI	Optimized UI with stable data
Manual `shouldComponentUpdate`	Optional	Built-in

CSR vs SSR

- Client-side rendering (CSR): creating dynamic websites by programmatically generating HTML on the client browser (DOM manipulation)
- Server-Side Rendering (SSR): the server generates the HTML for each route on the server, sends it to the browser, and React hydrates it to make it interactive



CSR	SSR
Page built **in the browser**	Page built **on the server**
HTML empty → JS fills it	HTML fully rendered → JS hydrates
Slow first load	Fast first load
SEO requires tricks	SEO works naturally

Additional materials

- Babel
- Webpack
- React 18 new features
- class component

Homework - Requirement

- ❏ 全程recording: 八股闭眼 & 摘耳机，无AI辅助
- ❏ 全程recording: Coding可适当AI辅助，但需写完(边解释边写)，模拟真实面试情景
- ❏ 录音提交地址:

  https://drive.google.com/drive/folders/1Mrm341uOIS8c2Ty6NwYvvSybHa6hESem?usp=drive_link
- ❏ 提交格式: 小组+日期，例如Group1_12/07/2025
- ❏ 提交截止时间: due第二天5PM (周四作业递延到下周一)

Homework - Class Code

❑ Write the code as taught in class, take a screenshot of your code, and post it in the Wechat group. **It’s due the same night.**

Homework - Short Answer Questions

1. Why would you use React? Name some advantages.
2. Name some differences between React and Angular.
3. Explain component-based architecture.
4. What is a SPA?
5. What is JSX?
6. What is state, and how do you update it? Can you mutate it directly?
7. What is the virtual DOM?
8. Explain the diffing algorithm? How do keys play a role?
9. Why should we not update the state directly?
10. What is the difference between Shadow DOM and Virtual DOM?

Homework - Short Answer Questions

11. What are fragments? Why fragments are better than container divs?
12. What is the recommended way for naming components?
13. What are the new changes in react 18?
14. How do prop updates affect rendering?
15. Explain the React component lifecycle and its methods.
16. What is the difference between a controlled component and uncontrolled component?
17. What is the difference between state and props?
18. What is "key" prop and what is the benefit of using it in arrays of elements?
19. Why you can't update props in React?
20. What are React 18 new features

Homework - Coding Questions

Leetcode:

283. Move Zeroes

217. Contains Duplicate

349. Intersection of Two Arrays

242. Valid Anagram

205. Isomorphic Strings

290. Word Pattern