

2

Typescript trong React

- React up-to-date
- Setup môi trường
- Typescript trong React
- Rendering & Life Cycle

React up-to-date

React 16 - 2017

- Fragment và string return type
- componentDidCatch error boundary
- Portal để render ngoài root node (modals)
- SSR improvement (link ở cuối bài)
- attributes mà React không handle sẽ pass thẳng vào DOM thay vì ignore
- “Fiber” core architecture (link ở cuối bài)
- Ngưng support IE < 11 do React sử dụng Map và Set, có thể polyfill

React 17 - 2020

- Không có feature mới
- Cho phép có nhiều version của React trong 1 dự án
- Không còn gắn eventListener trên DOM mà chuyển sang root node
- Đa phần là optimize react:
 - Có thể sử dụng event object trực tiếp từ handler mà không cần persist
 - useEffect cleanup sẽ chạy bất đồng bộ => không block unmount
 - Chỉnh sửa onScroll, onBlur và onFocus để đúng với trình duyệt
 - Và một số lỗi khác (link cuối bài)

React 18 - beta

- Tự động Batch các render lại với nhau
- Giới thiệu một số kỹ thuật mới để cải thiện performance
- Đổi React.render thành createRoot để tạo root 1 lần
- Hỗ trợ thêm cho SSR và streaming
- React 18 discussion group: <https://github.com/reactwg/react-18/discussions>

Setup môi trường

Requirements

- Node, Yarn, Git
- VS Code hay bất kỳ editor nào:
 - Cài support cho Typescript
 - Path Intellisense
 - esLint

```
TS TypeScript (typescript) - Configured Language
TS TypeScript React (typescriptreact)
```



Cài đặt

- yarn create react-app <tên-app> --template typescript
- yarn add react-router-dom @types/react-router-dom
- yarn start







```
"@types/jest": "^26.0.15",  
"@types/node": "^12.0.0",  
"@types/react": "^17.0.0",  
"@types/react-dom": "^17.0.0",  
"react": "^17.0.2",  
"react-dom": "^17.0.2",  
"react-scripts": "4.0.3",  
"typescript": "^4.1.2",
```


Typescript in action

Typescript trong HTML

Mặc định thì DOM API đã được support khi mình cài Typescript

```
document.body.getE
```

 getElementsByClassName	(method) Element.get
 getElementsByName	ElementsByName
 getElementsByTagNameNS	(classNames: strin
 getClientRects	g): HTMLCollectionOf
 getRootNode	<Element>
 getAttribute	Returns a HTMLCollection of the elements in the object on

HTML attributes type cũng được định sẵn khi mình nhờ @types/react-dom

```
default
f
fr
op. index.d.ts(1836, 9): The expected type comes from property 'className' which is declared here on type
'DetailedHTMLProps<HTMLAttributes<HTMLElement>, HTMLElement>'
()
(JSX attribute) React.HTMLAttributes<HTMLElement>.className?: string | undefined
View Problem No quick fixes available
er className={1}>
src={logo} className="App-logo" alt="logo" />
it <code>src/App.tsx</code> and save to reload.
className="App-link"
```

Typescript trong React

PropTypes

```
import PropTypes from 'prop-types'

const Card = ({
  title,
  rating,
  onClick,
  children
}) => (
  <div onClick={onClick}>
    <h4>{title}</h4>
    {rating ?? <h5>Rating: {rating}</h5>}
    <p>{children}</p>
  </div>
)

Card.propTypes = {
  title: PropTypes.string.isRequired,
  rating: PropTypes.number,
  onClick: PropTypes.func
}

export default Card
```

```
import { FC } from 'react'

type Props = {
  title: string
  rating?: number
  onClick?: () => void
}

const Card: FC<Props> = ({
  title,
  rating,
  onClick,
  children
}) => (
  <div onClick={onClick}>
    <h4>{title}</h4>
    {rating ?? <h5>Rating: {rating}</h5>}
    <p>{children}</p>
  </div>
)

export default Card
```

Typescript trong React

PropTypes (tiếp)

```
import PropTypes from 'prop-types'  
  
const Link = ({  
  color = 'blue',  
  children,  
  style,  
  ...restProps  
}) => (  
  <a {...restProps} style={{ color, ...style }}>  
    {children}  
  </a>  
)  
  
Link.propTypes = {  
  color: PropTypes.oneOf(['red', 'green', 'blue'])  
}  
  
export default Link
```

```
import { FC, HTMLProps } from 'react'  
  
type Props = {  
  color?: 'red' | 'green' | 'blue'  
} & HTMLProps<HTMLAnchorElement>  
  
const Link: FC<Props> = ({  
  color = 'blue',  
  children,  
  style,  
  ...restProps  
}) => (  
  <a {...restProps} style={{ color, ...style }}>  
    {children}  
  </a>  
)  
  
export default Link
```

Typescript trong React

State

```
const [expanded, setExpanded] = useState(false)

const [expanded, setExpanded] = useState<boolean | null>(null)
```

```
type ComplexState = {
  expanded: boolean
  priority: 'high' | 'low'
}

const [state, setState] = useState<ComplexState | null>(null)
```

Typescript trong React

Event handlers

```
onClick={() => { console.log(event.target) } }>
```

```
(property) React.BaseSyntheticEvent<globalThis.MouseEvent, EventTarget & HTMLDivElement, EventTarget>.target: EventTarget
```



target



currentTarget



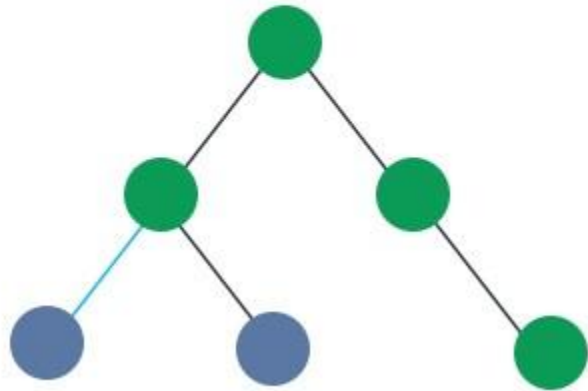
relatedTarget

```
onClick?: (e: MouseEvent<HTMLDivElement>) => void
```

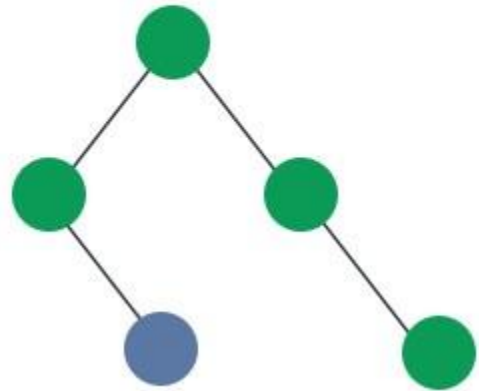
Life cycle & virtual DOM

DOM (Document Object Model)

Virtual DOM

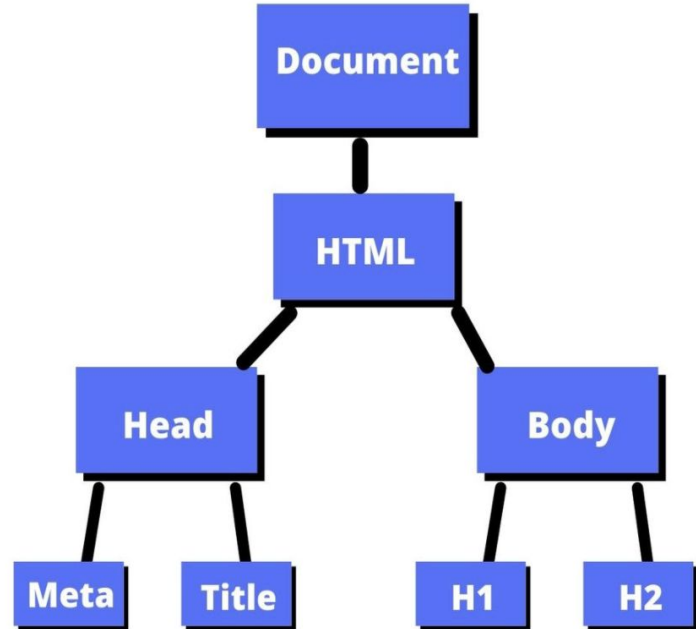


Real DOM



Real DOM

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>DOM tree structure</title>
  </head>
  <body>
    <h1>DOM tree structure</h1>
    <h2>Learn about the DOM</h2>
  </body>
</html>
```



Virtual DOM

```
const element = (  
  <div className="container">  
    <a>This is a element</a>  
  </div>  
)
```




```
{  
  $$typeof: Symbol(react.element), type: 'div', key: null, ref: null, props: {  
    ...  
  }, ...  
}  
$$typeof: Symbol(react.element)  
key: null  
props:  
children:  
  $$typeof: Symbol(react.element)  
  key: null  
  props:  
    children: "This is a element"  
    [[Prototype]]: Object  
  ref: null  
  type: "a"  
  _owner: null  
  _store: {validated: true}  
  _self: undefined  
  _source: {fileName: '/online-course/pages/index.js', lineNumber: 26, columnNumber: 7}  
  [[Prototype]]: Object  
  className: "container"  
  [[Prototype]]: Object  
  ref: null  
  type: "div"  
  _owner: null  
  _store: {validated: false}  
  _self: undefined  
  _source: {fileName: '/online-course/pages/index.js', lineNumber: 25, columnNumber: 5}  
  [[Prototype]]: Object  
}
```

Rendering Elements

React Element

- Là một **Object** (đối tượng) mô tả một **HTML Element**.
- Được gọi từ **Function** hoặc **Class Component**.
- Không thể thấy được ở Browser (trình duyệt).



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

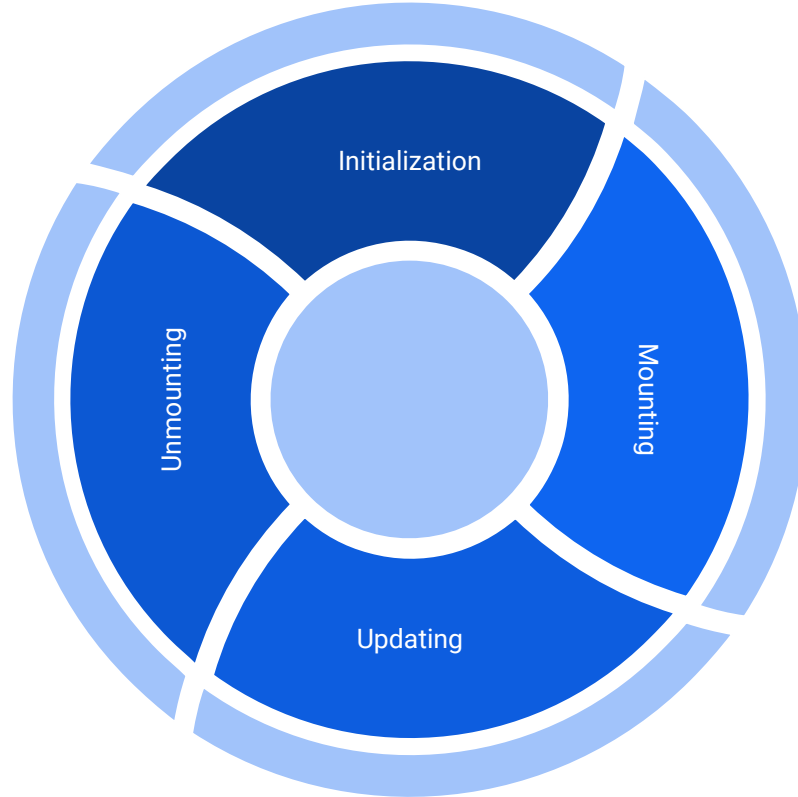
Rendering Elements

React Component

- Giống như một function trong Javascript. Có Input (props) và Output (React Element)
- Dùng để phân chia các UI thành các phần nhỏ để dễ dàng quản lý
- Có 2 cách để dùng Component là **Function** và **Class Component**

```
function Comment(props) {  
  return (  
    <div className="Comment">  
      <div className="UserInfo">  
        <img className="Avatar"  
          src={props.author.avatarUrl}  
          alt={props.author.name}  
        />  
        <div className="UserInfo-name">  
          {props.author.name}  
        </div>  
      </div>  
      <div className="Comment-text">  
        {props.text}  
      </div>  
      <div className="Comment-date">  
        {formatDate(props.date)}  
      </div>  
    </div>  
  );  
}
```

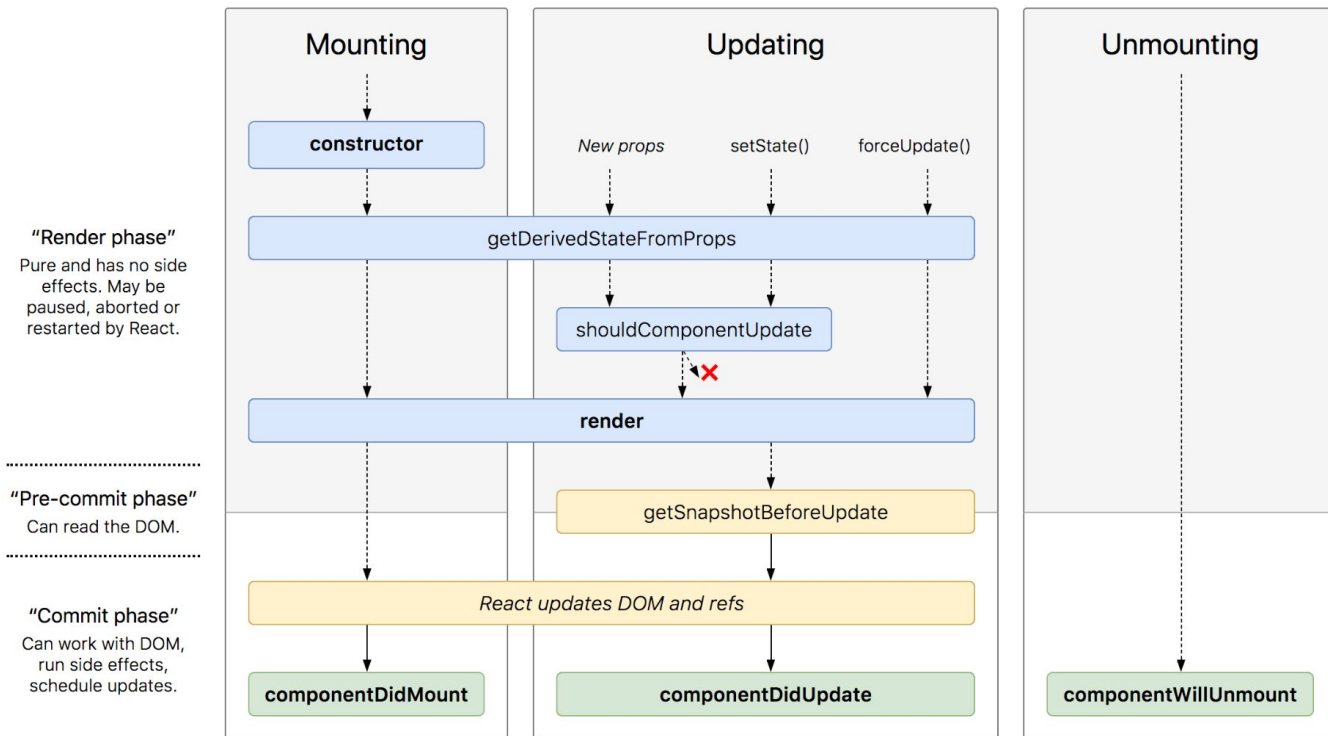
Lifecycle of Component



Lifecycle of Component

React version 16.4

Language en-US



Bài tập

Thêm Typescript cho TodoApp

https://github.com/JeelGajera/React_Todo-app

Để thêm typescript vào 1 dự án React:

```
yarn add typescript @types/node @types/react @types/react-dom
```

Link tham khảo

Event handler: <https://www.carlrippon.com/React-event-handlers-with-typescript/>

SSR improvement: <https://medium.com/@aickin/whats-new-with-server-side-rendering-in-react-16-9b0d78585d67>

Fiber: <https://engineering.fb.com/2017/09/26/web/react-16-a-look-inside-an-api-compatible-rewrite-of-our-frontend-ui-library/>

React 17: <https://reactjs.org/blog/2020/10/20/react-v17.html>