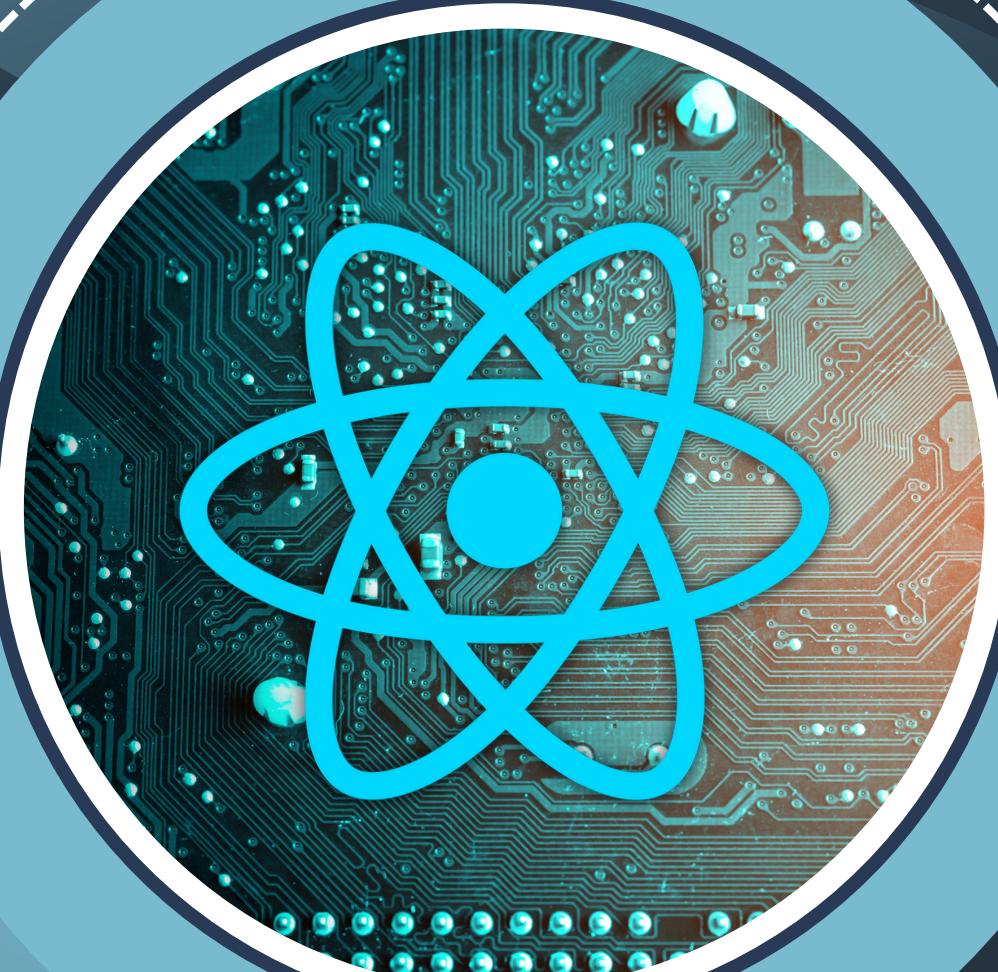


Singleton

دیزاین پترن هادر React

Creational Patterns

به زبان ساده همراه با مثال های عملی
کاربرد دیزاین پترن ها برای برنامه نویسان React



Creational Patterns

Singleton

”

الگوی Singleton تضمین می کند که یک کلاس فقط یک نمونه (Instance) داشته باشد و دسترسی به آن نمونه از هر جای برنامه ممکن باشد. از این الگو می توانیم برای مدیریت داده های مشترک، ذخیره تنظیمات یا ارتباط بین اجزای مختلف برنامه استفاده کنیم.

فرض کنید برای یک فروشگاه می خواهیم یک Singleton بسازیم که سفارشات یک کاربر را مدیریت کند و این داده ها بین چند کامپونت مانند بخش ایجاد سفارش جدید یا نمایش لیست سفارشات به اشتراک گذاشته شود.

بخش Order Interface

یک Interface زیر می سازیم.



Order



```
export interface Order {  
  id: number;  
  qty: number;  
}
```

بخش OrderService

اگر نمونه ای از OrderService قبل ساخته شده باشد، همان نمونه بازگردانده می شود.
از جلوگیری برای تغییر شیء استفاده می شود.



```
● ● ●  
class OrderService {  
    static instance: OrderService;  
    data: Order[] = [];  
  
    constructor() {  
        if (!OrderService.instance) {  
            this.data = [];  
            OrderService.instance = this;  
        }  
        return OrderService.instance;  
    }  
  
    set(order: Order) {  
        this.data.push(order);  
    }  
  
    get() {  
        return this.data;  
    }  
}
```

```
● ● ●  
const service = new OrderService();  
Object.freeze(service);  
export default service;
```

بخش CreateOrder

یک Component برای ایجاد سفارش جدید به شکل زیر می سازیم.



```
import service from "../pattern/OrderService";

const CreateOrder = () => {
  const handleClick = () => {
    service.set({ id: 1992, qty: 1 });
  };

  return <button onClick={handleClick}>Create</button>;
};

export default CreateOrder;
```

بخش ReadOrder

یک Component برای نمایش لیست سفارشات به شکل زیر می سازیم.



```
● ● ●  
  
import { useState } from "react";  
import service, { Order } from "../pattern/orderService";  
  
const ReadOrder = () => {  
  const [order, setOrder] = useState<Order[]>([]);  
  
  const handleClick = () => {  
    setOrder([...service.get()]);  
  };  
  
  return (  
    <div>  
      <button onClick={handleClick}>Read</button>  
      {order.length > 0 ? (  
        <ul>  
          {order.map((item, index) => (  
            <li key={index}>  
              ID: {item.id}, Quantity: {item.qty}  
            </li>  
          ))}  
        </ul>  
      ) : (  
        <p>Empty!</p>  
      )}  
    </div>  
  );  
};  
  
export default ReadOrder;
```

```
● ● ●  
  
<div>  
  <button onClick={handleClick}>Read</button>  
  {order.length > 0 ? (  
    <ul>  
      {order.map((item, index) => (  
        <li key={index}>  
          ID: {item.id}, Quantity: {item.qty}  
        </li>  
      ))}  
    </ul>  
  ) : (  
    <p>Empty!</p>  
  )}  
</div>
```

طرز استفاده از Component‌ها

در پایان به شکل زیر از Component‌هایی که برای ایجاد سفارش جدید و نمایش لیست سفارشات ساختیم، استفاده می‌کنیم.



SingletonDesignPattern

```
import CreateOrder from "./components/CreateOrder";
import ReadOrder from "./components/ReadOrder";

const SingletonDesignPattern = () => {
  return (
    <div>
      <CreateOrder/>
      <ReadOrder/>
    </div>
  );
};

export default SingletonDesignPattern;
```

