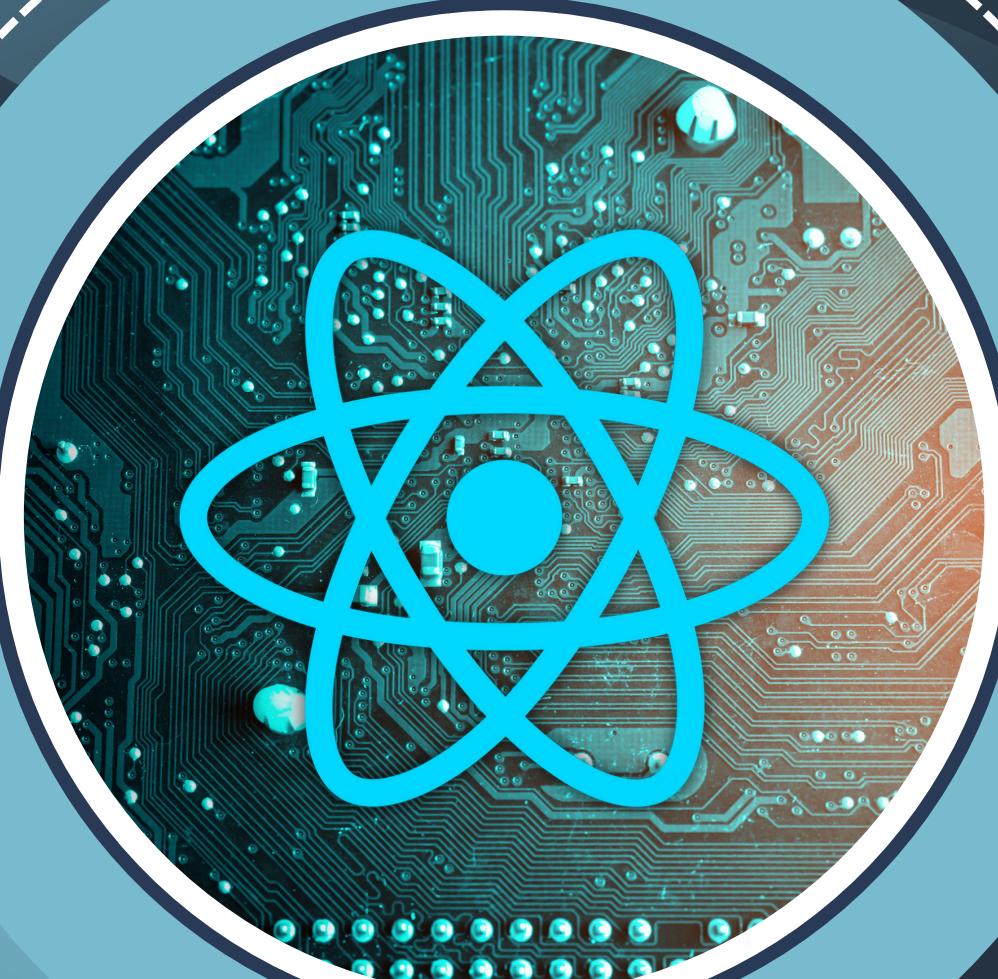


Abstract Factory

دیزاین پترن هادر React

Creational Patterns

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



Creational Patterns

Abstract Factory



الگوی **Abstract Factory** یعنی ساخت یک **Factory** که خودش چندین **Factory** دیگر می‌سازد. هر **Factory**، اجزای مرتبط با یک موضوع یا سبک خاص تولید می‌کند، بدون اینکه لازم باشد کسی که از این اجزا استفاده می‌کند بداند این اجزا دقیقاً چگونه ساخته شده‌اند.

برای درک راحت تر واژه **Factory** فرض کنید می‌خواهیم یک بخش داشته باشیم که وظیفه هندل کردن **Theme** را بر عینده دارد، به این بخش می‌گوییم **Factory** و دو **Factory** دیگر که یکی وظیفه تم **Light** و دیگری وظیفه تم **Dark** را بر عینده دارند، زیر مجموعه این **Factory** هستند.

بخش Theme Factory

یک Interface که دارای دو Method برای ساخت Component مورد نظر می باشد.



Theme Factor



```
export interface IThemeFactory {  
    createButton(): React.ReactNode;  
    createInput(): React.ReactNode;  
}
```

ایجاد Light Factory برای تم

در این بخش Component های مورد نظر را با تم Light پیاده سازی می کنیم.



LightThemeFactory



```
import { IThemeFactory } from "./ThemeFactory.interface";

export class LightThemeFactory implements IThemeFactory {
    createButton(): React.ReactNode {
        return <button style={{ backgroundColor: "white", color: "black" }}>Light Button</button>;
    }

    createInput(): React.ReactNode {
        return <input style={{ backgroundColor: "white", color: "black", border: "1px solid black" }} />;
    }
}
```

ایجاد Factory برای تم Dark

در این بخش Component های مورد نظر را با تم Dark پیاده سازی می کنیم.



DarkThemeFactory

```
● ● ●

import { IThemeFactory } from "./ThemeFactory.interface";

export class DarkThemeFactory implements IThemeFactory {
    createButton(): React.ReactNode {
        return <button style={{ backgroundColor: "black", color: "white" }}>Dark Button</button>;
    }

    createInput(): React.ReactNode {
        return <input style={{ backgroundColor: "black", color: "white", border: "1px solid white" }} />;
    }
}
```

ایجاد Factory اصلی

این Factory همان Factory اصلی است که دو زیر مجموعه آن هستند.



ThemeComponent

```
import { IThemeFactory } from "./ThemeFactory.interface";
import { LightThemeFactory } from "./LightThemeFactory";
import { DarkThemeFactory } from "./DarkThemeFactory";

type ThemeType = "light" | "dark";

interface ThemeComponentProps {
  theme: ThemeType;
}

const ThemeComponent: React.FC<ThemeComponentProps> = ({ theme }) => {
  let factory: IThemeFactory;

  if (theme === "light") {
    factory = new LightThemeFactory();
  } else {
    factory = new DarkThemeFactory();
  }

  return (
    <div>
      {factory.createButton()}
      {factory.createInput()}
    </div>
  );
};

export default ThemeComponent;
```

ThemeComponent

LightThemeFactory

DarkThemeFactory



طرز استفاده از Factory اصلی

در پایان به شکل زیر از Factory که ساختیم، استفاده می کنیم.



ThemeComponent

```
● ● ●

import React, { useState } from "react";
import ThemeComponent from "./abstract-factory/ThemeComponent";

const AbstractFactoryDesignPattern: React.FC = () => {
  const [theme, setTheme] = useState<"light" | "dark">("light");

  const toggleTheme = () => {
    setTheme((prev) => (prev === "light" ? "dark" : "light"));
  };

  return (
    <div>
      <button onClick={toggleTheme}>Toggle Theme</button>
      <ThemeComponent theme={theme} />
    </div>
  );
};

export default AbstractFactoryDesignPattern;
```