

# Factory Method

دیزاین پترن ها در React

Creational Patterns

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



# Creational Patterns

## Factory Method

”

الگوی Factory Method به ما این امکان را می دهد که ایجاد یک شیء یا کامپوننت را به کلاس های فرزند یا یک متده شخص بسپاریم، بدون این که نیاز باشد منطق دقیق ساخت آن ها را در سراسر کد تکرار کنیم. می توانیم برای ایجاد کامپوننت های تکراری بارفته های مختلف از آن استفاده کنیم.

فرض کنید چند نوع دکمه داریم؛ مانند دکمه Primary و Secondary. هر بار باید بصورت دستی تصمیم بگیریم که کدام دکمه را بنویسیم. اینجاست که این الگو به ما کمک می کند. این روش یک تابع می سازد که خودش تصمیم می گیرد که کدام دکمه را بر اساس نوع (type) برگرداند.

# بخش ButtonProps

ابتدا Props مربوط به Component را می سازیم.



## ButtonProps



```
export interface ButtonProps {  
  label: string;  
  onClick: () => void;  
}
```

# بخش Button

در این بخش Component مد نظر خود را پیاده سازی می کنیم.



## Button

```
import { ButtonProps } from "./Button.props";

export const PrimaryButton: React.FC<ButtonProps> = ({ label, onClick }) => (
  <button style={{ backgroundColor: "blue", color: "white" }} onClick={onClick}>
    {label}
  </button>
);

export const SecondaryButton: React.FC<ButtonProps> = ({ label, onClick }) => (
  <button style={{ backgroundColor: "gray", color: "black" }} onClick={onClick}>
    {label}
  </button>
);
```

# بخش ButtonFactory

در این بخش بر اساس Component Type ورودی، مورد نظر برگردانده می شود.

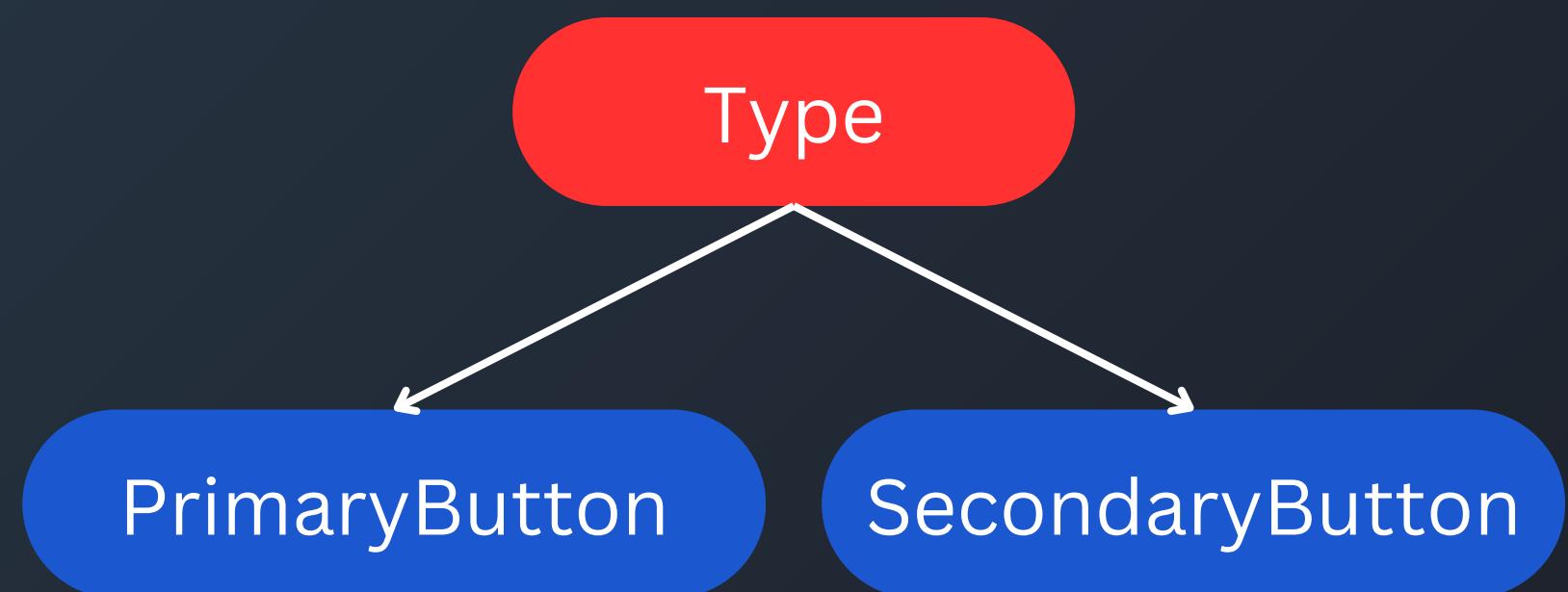


```
● ● ●

import { PrimaryButton } from "./Button";
import { SecondaryButton } from "./Button";
import { ButtonProps } from "./Button.props";

type ButtonType = "primary" | "secondary";

export function createButton(type: ButtonType): React.FC<ButtonProps> {
  switch (type) {
    case "primary":
      return PrimaryButton;
    case "secondary":
      return SecondaryButton;
    default:
      throw new Error(`Unknown button type: ${type}`);
  }
}
```



# طرز استفاده از ButtonFactory

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



## FactoryMethodDesignPattern

```
import { createButton } from "./pattern/ButtonFactory";

const FactoryMethodDesignPattern: React.FC = () => {
  const PrimaryButton = createButton("primary");
  const SecondaryButton = createButton("secondary");

  return (
    <div>
      <PrimaryButton
        label="Click Me"
        onClick={() => alert("Primary Button Clicked")}
      />
      <SecondaryButton
        label="Cancel"
        onClick={() => alert("Secondary Button Clicked")}
      />
    </div>
  );
};

export default FactoryMethodDesignPattern;
```