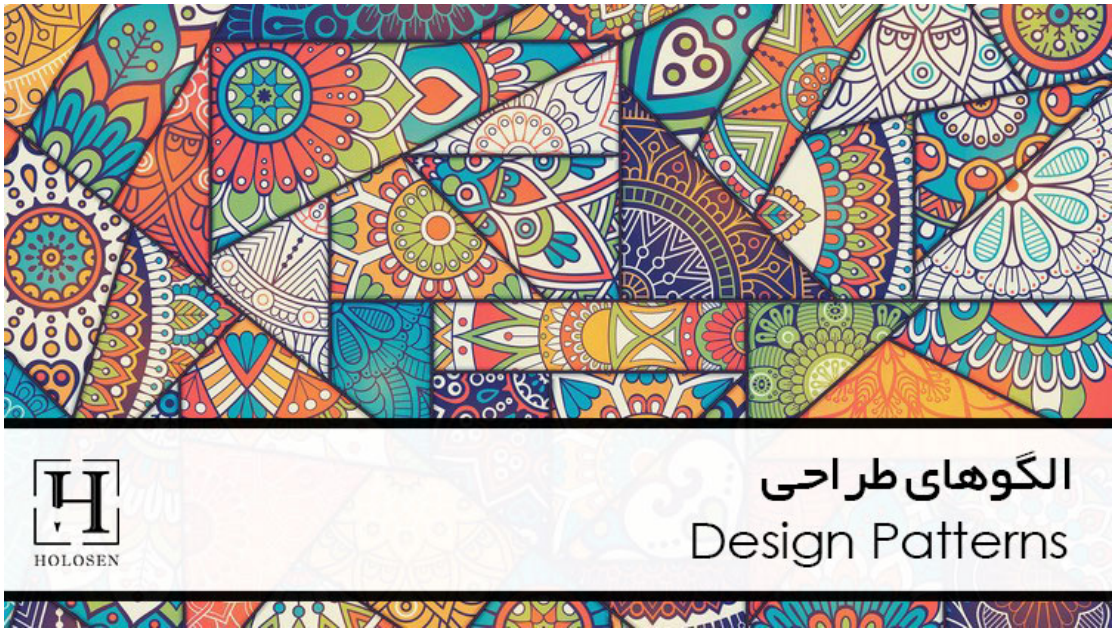


دیزاین پترن BUILDER

CREATIONAL DESIGN PATTERNS

یکی از زیرشاخه های الگوهای طراحی، Creational Design Patterns است. این الگو راهی جهت ساخت و ایجاد اشیاء (object) از کلاس ها را ارائه میدهد. ساخت یک شیء با استفاده از new کردن آن کلاس در اصطلاح hard code کردن راه حل خوبی نیست و بهتر است از الگوهای طراحی Creational استفاده کرد.

BUILDER DESIGN PATTERN

این پترن یکی از زیرشاخه های الگوهای طراحی از نوع Creational است.

از دیزاین پترن builder یا سازنده زمانی استفاده میشود که بخواهیم یک شیء با پیچیدگی ها و تنظیمات custom زیاد را بسازیم.

نحوه کارکرد این دیزاین پترن به این صورت است که زمانی که شیئی را ساختیم تنظیماتی که میخواهیم روی شیء اعمال کنیم یا مقادیر متغیرهارا پر کنیم، با استفاده از متدهای مربوطه این کار را انجام میدهیم.

میزان استفاده : 1 2 3 4 5 متوسط رو به پایین

استفاده از این دیزاین پترن موجب کاهش پیچیدگی ها در ساخت اشیاء میشود.

فرض کنید میخواهیم پیتزا درست کنیم. یک کلاس پیتزا بصورت زیر مینویسیم.

```
internal class Pizza
{
    public string Name { get; set; }
    public bool Spicy { get; set; }
    public bool Sauce { get; set; }
    public bool Cheese { get; set; }

    public override string ToString()
    {
        string result = "";
        if (!string.IsNullOrEmpty(Name) && !string.IsNullOrWhiteSpace(
            result += Name + "\n";
        result += Cheese ? "Has Cheese\n" : "";
        result += Sauce ? "Has Sauce\n" : "";
        result += Spicy ? "Is Spicy\n" : "";
        return result;
    }
}
```

مشخصات این پیتزا میتواند مقادیر فلفلی (spicy)، همراه با سس (sauce) و همراه با پنیر (cheese) داشته باشد.

حالا کلاس اصلی سازنده پیتزا را که وظیفه پخت پیتزا را دارد با نام PizzaBuilder بصورت زیر مینویسیم.

```
internal class PizzaBuilder
{
    private Pizza pizza;
    public PizzaBuilder()
    {
        pizza = new Pizza();
    }
    public PizzaBuilder SetName(string name)
    {
        pizza.Name = name;
        return this;
    }
    public PizzaBuilder AddSauce()
    {
        pizza.Sauce = true;
        return this;
    }
    public PizzaBuilder AddPepperoni()
    {
        pizza.Spicy = true;
        return this;
    }
    public PizzaBuilder AddCheese()
    {
        pizza.Cheese = true;
        return this;
    }
    public Pizza Bake()
    {
        return pizza;
    }
}
```

کلاس PizzaBuilder دارای متدی جهت اضافه کردن سس (AddSauce)، متدی جهت اضافه کردن پپرونی جهت تند کردن (AddPepperoni)، متدی جهت اضافه کردن پنیر (AddCheese)

و متدی جهت پخت (Bake) دارد.

نکته مهم در رابطه با این متدها این است که مقدار بازگشتی متدهای کمکی از نوع خود PizzaBuilder است که همین کلاسی است که درحال آماده سازی است. در نهایت متد پخت از نوع پیتزا به ما برمیگرداند.

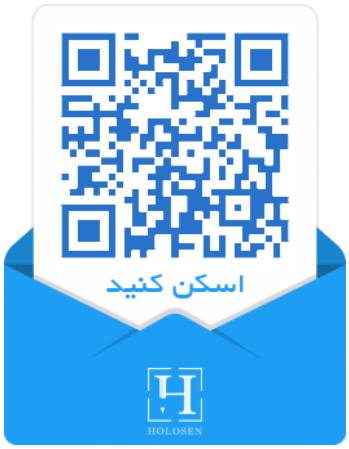
حال به سراغ پخت پیتزا در متد main میپردازیم.

```
static void Main(string[] args)
{
    try
    {
        var pizza = new PizzaBuilder()
            .SetName("Peperony")
            .AddCheese()
            .AddPepperoni()
            .AddSauce().Bake();
        Console.WriteLine(pizza.ToString());
    }
    catch (Exception ex)
    {
        ShowError(ex.Message);
    }
    Console.ReadLine();
}
```

در این متد میتوانیم بصورت متوالی بعد از ایجاد شیء از PizzaBuilder متدهای AddPepperoni، AddCheese و AddSauce را صدا بزنیم سپس اقدام به پخت (Bake) بکنیم.

خروجی این برنامه بصورت زیر نمایش داده خواهد شد.

```
Peperony
Has Cheese
Has Sauce
Is Spicy
```



هولوسن
با من یاد بگیر

آموزش های بیشتر در وبسایت هولوسن : https://holosen.net