Wzorzec Projektowy Adapter

Czym Jest Adapter?

Adapter jest strukturalnym wzorcem projektowym pozwalającym na współdziałanie ze sobą obiektów o niekompatybilnych interfejsach.

Na czym polega?

Jest to połączenie dwóch niekompatybilnych interfejsów, tak aby mogły ze sobą współpracować. Adapter działa jak tłumacz pomiędzy dwoma klasami — opakowuje obiekt jednej klasy i udostępnia jego funkcjonalność w sposób zgodny z oczekiwanym przez drugą klasę interfejsem.

Jaki problem rozwiazuje?

rozwiązuje problem w którym mamy dwie klasy, które powinny współpracować, ale nie mogą, bo mają inne interfejsy (np. inne metody, inne nazwy metod, inne parametry)

Przykład

- Tworzymy aplikacje Monitorującą giełdę
- Pobiera ona dane z wielu źródeł w formatach XML
- Wykorzystujemy te dane w naszej Aplikacji

Przykład cz.2

- Chcemy Ulepszyć naszą aplikacje dodając nową bibliotekę która działa tylko na danych w formacie JSON
- Nie można zastosować biblioteki analitycznej od razu, bo wymaga ona przedstawiania danych w formacie który jest niekompatybilny z aplikacją.
- można stworzyć adaptery XML-do-JSON dla każdej klasy biblioteki analitycznej, która jest używana przez nasz kod. Gdy adapter otrzyma wywołanie, tłumaczy przychodzące dane XML na strukturę JSON i przekazuje wywołanie dalej.

Interfejs Aplikacji

```
public interface IPaymentProcessor
{
    void Pay(double amount);
}
```

Wygląd zewnętrznej biblioteki – której musimy użyć, Interfejsy się nie zgadzają: system używa Pay(), a Stripe używa MakeStripePayment().

```
public class StripePayment
{
    public void MakeStripePayment(double total)
    {
        Console.WriteLine($"[Stripe] Paid {total} PLN");
    }
}
```

Adapter– implementuje nowy interfejs i używa starego obiektu

```
public class StripeAdapter : IPaymentProcessor
{
    private readonly StripePayment _stripe;

    public StripeAdapter(StripePayment stripe)
    {
        _stripe = stripe;
    }

    public void Pay(double amount)
    {
        _stripe.MakeStripePayment(amount);
    }
}
```

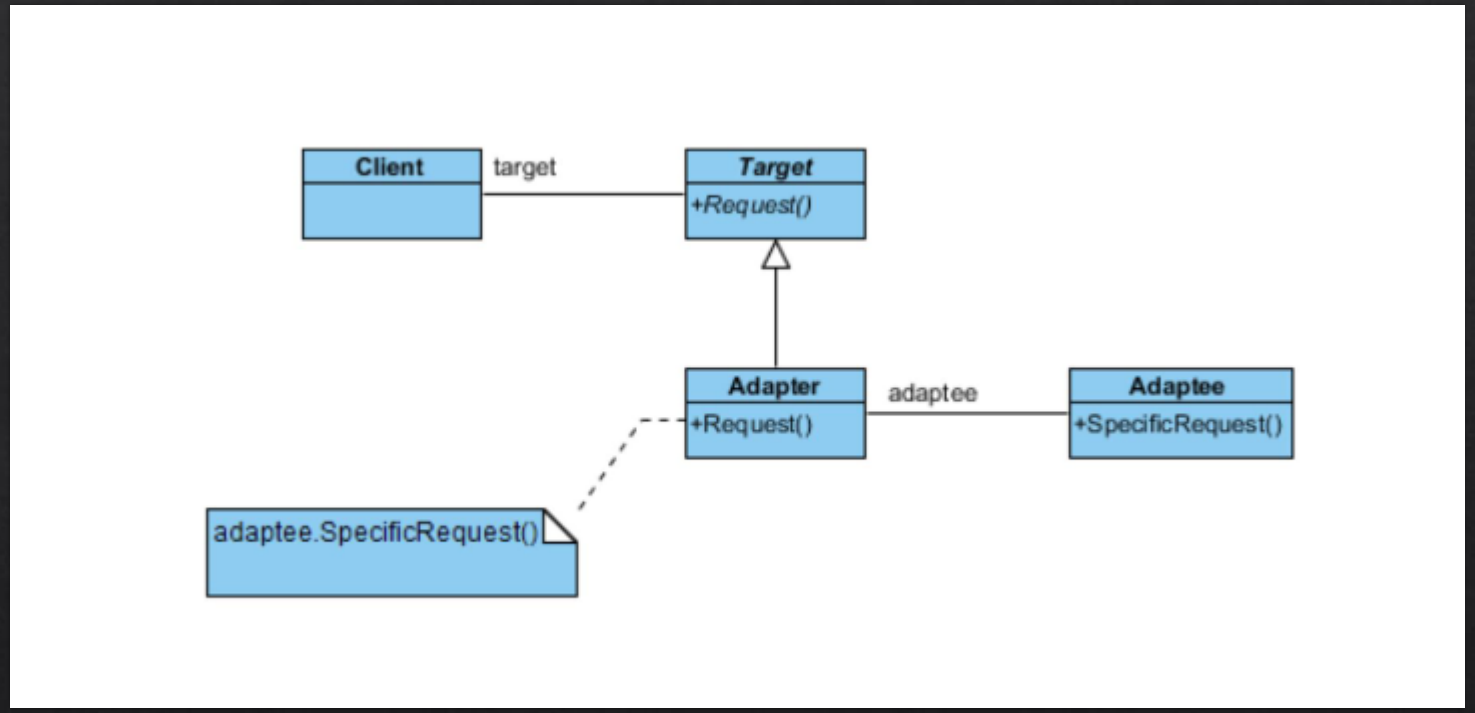
Wywołanie

```
class Program
{
    static void Main()
    {
        StripePayment stripe = new StripePayment();
        IPaymentProcessor paymentProcessor = new StripeAdapter(stripe);

        paymentProcessor.Pay(199.99);
    }
}
```

Konsola

```
[Stripe] Paid 199.99 PLN
```


Client
target
Target
+Request()
Adapter
+Request()
adaptee
Adaptee
+SpecificRequest()
adaptee.SpecificRequest()

Dziękuje za uwagę