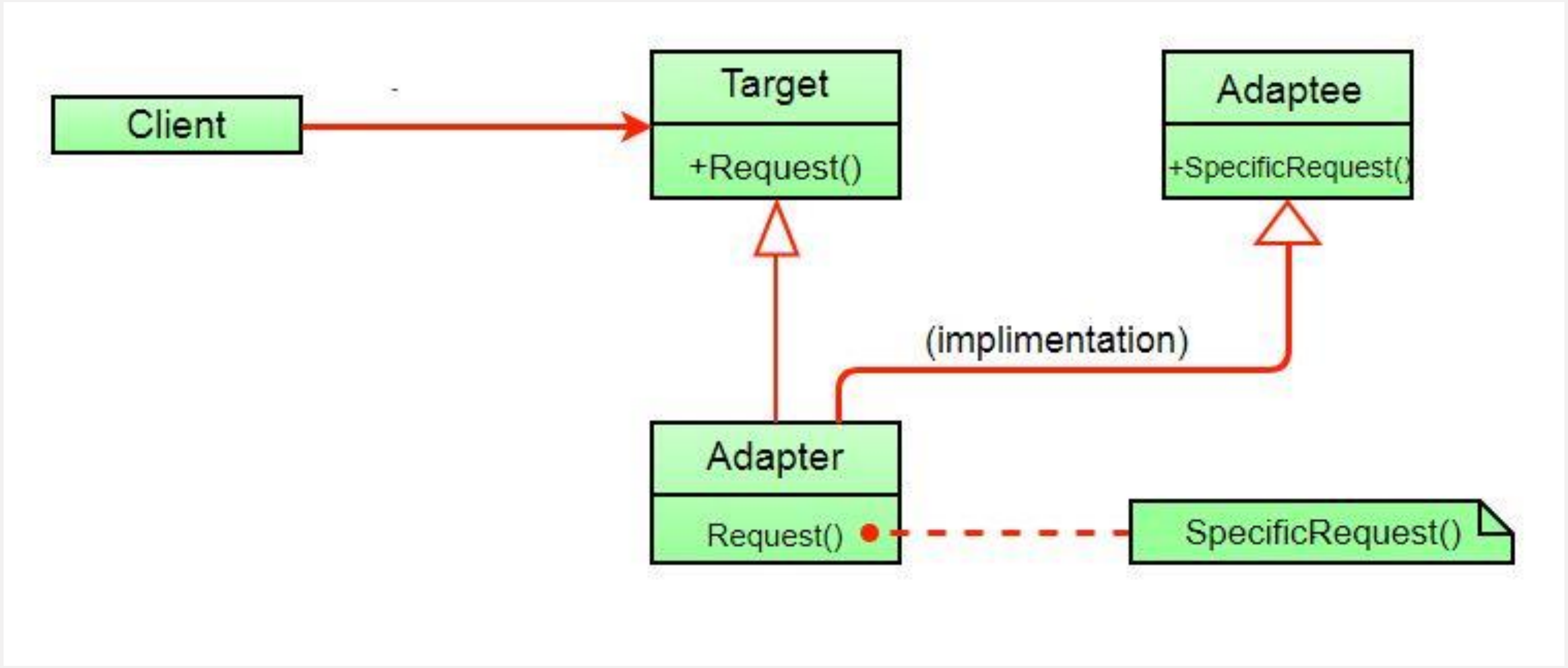
Adapter Pattern

Lecture-5

Adapter Pattern

- Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces
- Wrap an existing class with a new interface
- Impedance match an old component to a new system

Adapter Pattern Diagram



Scenario

I was working on a project where we had to call an API of an external system in order to calculate tax. The server expected information like zip code, product, price, quantity, etc to calculate the tax. But the bad thing was, it expected the data in XML format. XML and SOAP APIs are really really rare these days except in some banking applications or financial institutions. All our internal APIs expect JSON as input and return JSON as output. Now if we want to make a call to the external server, we have to convert the JSON data into XML format

Sample Code

```
class Json {
    public Json(){}
    Xml convertToXML(){
        // Logic to convert the data into Xml
    }
}
```

```
Json json = new Json("some json data");
Xml xml = json.convertToXml();
```

Implementation

```
class JsonToXmlAdapter implements IDataAdapter {
    // It contains an instance of an Adaptee
    private Json json;
    public JsonToXmlAdapter(Json json){
        this.json = json;
    }

    public Xml convert (){
        // Logic to convert Json to Xml
        this.json.convertToXML()
    }
}
```

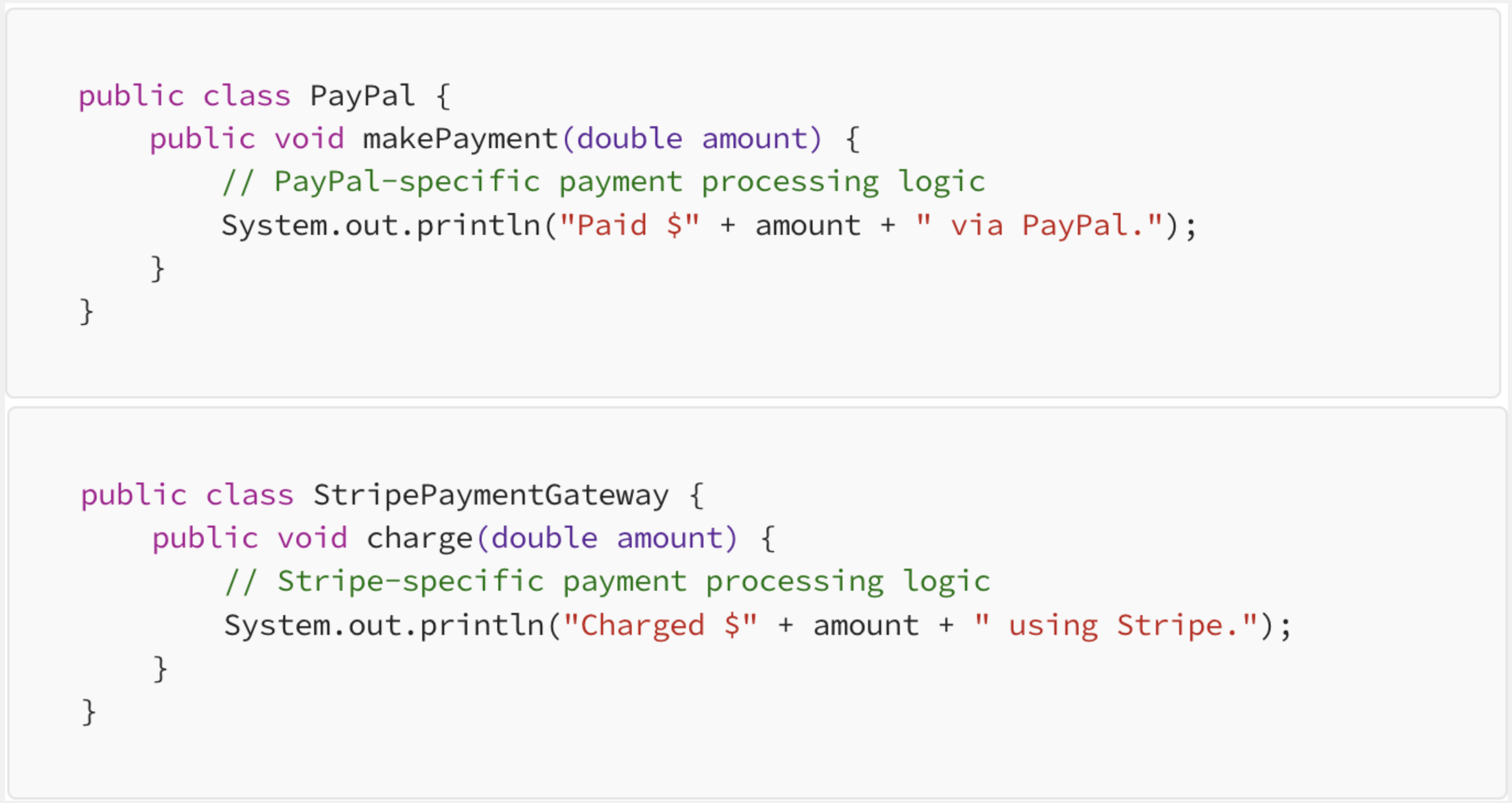
```
Json json = new Json("some json data");
IDataAdapter adapter = new JsonToXmlAdapter(json);
Xml xml = adapater.convert()
// Call the calculate tax API using the XML data
Decimal tax = calculateTax(xml);
```

```
// We have all kinds of different types of data
Json json = new Json("some json data");
Csv csv = new Csv("some csv data");

Bson bson = new Bson("some bson data");

// Client code

// Convert from Json to Xml
IDataAdapter adapter = new JsonToXmlAdapter(json);
Xml xml = adapter.convert();

// Convert Csv to Xml
adapter = new CsvToXmlAdapter(csv);
 xml = adapter.convert();

// Convert Bson to Xml
adapter = new BsonToXmlAdapter(bson);
 xml = adapter.convert();
```

Scenario

Imagine you’re building an e-commerce application that supports payments from different payment gateway providers, such as PayPal and Stripe. Each payment gateway has its own unique interface and methods for processing payments. However, you want to create a unified payment processing system in your application that can work with various payment gateway providers without changing your existing code

Sample Code



```java
public class PayPal {
    public void makePayment(double amount) {
        // PayPal-specific payment processing logic
        System.out.println("Paid $" + amount + " via PayPal.");
    }
}
```

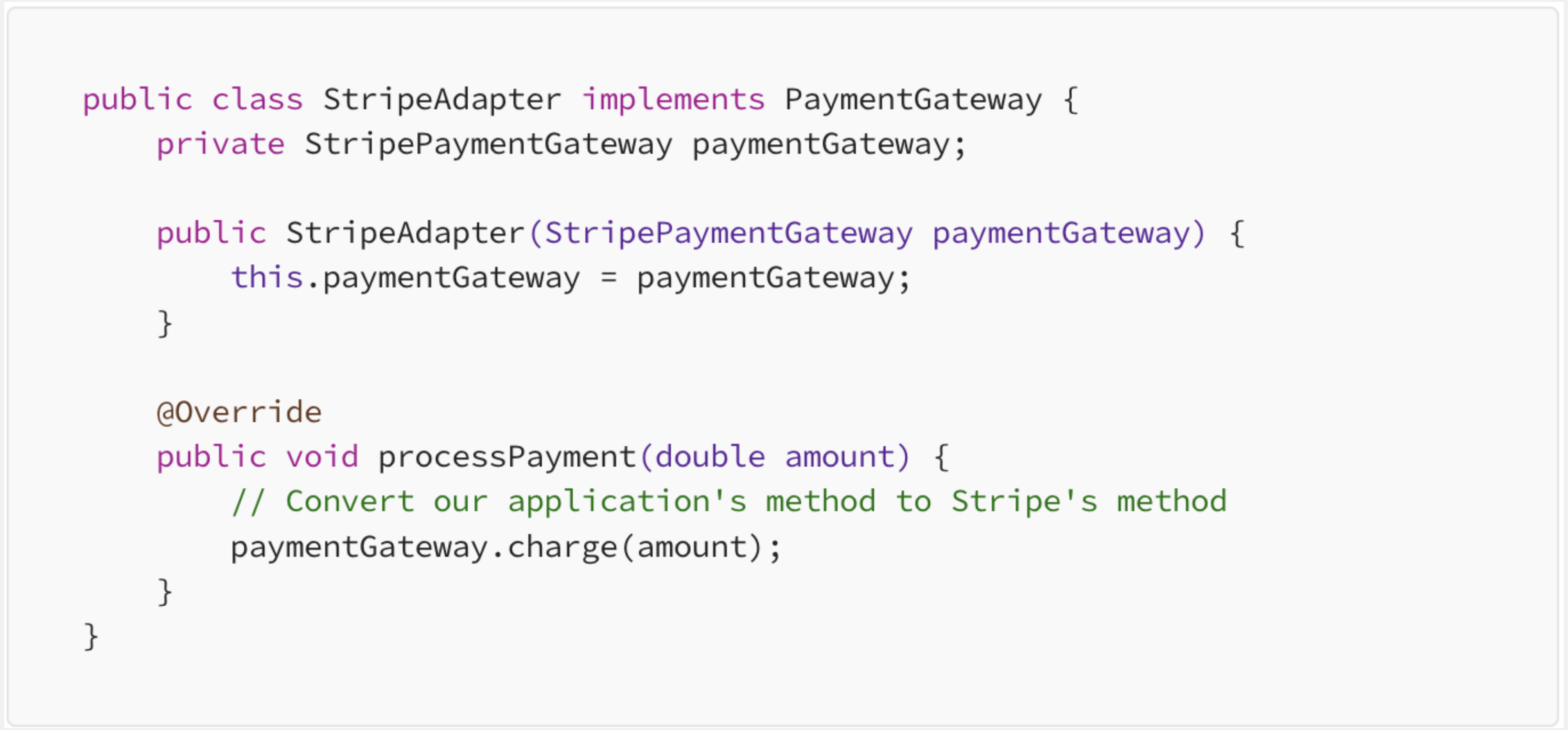
```java
public class StripePaymentGateway {
    public void charge(double amount) {
        // Stripe-specific payment processing logic
        System.out.println("Charged $" + amount + " using Stripe.");
    }
}
```

Sample Code

```java
public interface PaymentGateway {
    void processPayment(double amount);
}
```

```java
public class PayPalAdapter implements PaymentGateway {
    private PayPal paymentGateway;

    public PayPalAdapter(PayPal paymentGateway) {
        this.paymentGateway = paymentGateway;
    }

    @Override
    public void processPayment(double amount) {
        // Convert our application's method to PayPal's method
        paymentGateway.makePayment(amount);
    }
}
```

Sample Code



```java
public class StripeAdapter implements PaymentGateway {
    private StripePaymentGateway paymentGateway;

    public StripeAdapter(StripePaymentGateway paymentGateway) {
        this.paymentGateway = paymentGateway;
    }

    @Override
    public void processPayment(double amount) {
        // Convert our application's method to Stripe's method
        paymentGateway.charge(amount);
    }
}
```

Sample Code

```java
public class PaymentApp {
    public static void main(String[] args) {
        PaymentGateway paypalGateway = new PayPalAdapter(new PayPal());
        PaymentGateway stripeGateway = new StripeAdapter(new StripePaymentGateway())

        double amount = 100.0;

        // Process payments using different payment gateways
        paypalGateway.processPayment(amount);
        stripeGateway.processPayment(amount);
    }
}
```