Adapter Pattern

An Adapter Pattern says that just **"converts the interface of a class into another interface that a client wants".**

In other words, to provide the interface according to client requirement while using the services of a class with a different interface.

The Adapter Pattern is also known as **Wrapper.**

Advantage of Adapter Pattern

- It allows two or more previously incompatible objects to interact.
- It allows reusability of existing functionality.

Usage of Adapter pattern:

It is used:

- When an object needs to utilize an existing class with an incompatible interface.
- When you want to create a reusable class that cooperates with classes which don't have compatible interfaces.
- When you want to create a reusable class that cooperates with classes which don't have compatible interfaces.

Example of Adapter Pattern

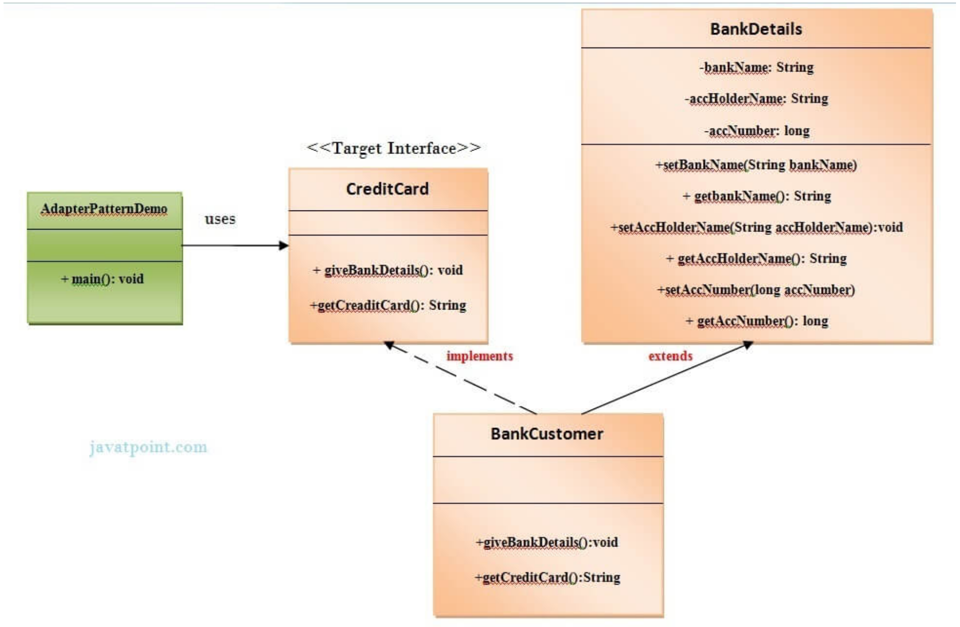
Let's understand the example of adapter design pattern by the above UML diagram.

UML for Adapter Pattern:

There are the following specifications for the adapter pattern:

- **Target Interface:** This is the desired interface class which will be used by the clients.

- **Adapter class:** This class is a wrapper class which implements the desired target interface and modifies the specific request available from the Adaptee class.
- **Adaptee class:** This is the class which is used by the Adapter class to reuse the existing functionality and modify them for desired use.
- **Client:** This class will interact with the Adapter class.



Implementation of above UML:

TWILIO SendGrid

Email Built For And By Developers

Customizable APIs
7 Language Libraries
Detailed Stats

Try for free

Step 1

Create a **CreditCard** interface (Target interface).

```
public interface CreditCard {
    public void giveBankDetails();
    public String getCreditCard();
}// End of the CreditCard interface.
```

Step 2

Create a **BankDetails** class (Adaptee class).

File: BankDetails.java

```
// This is the adapter class.
public class BankDetails{
    private String bankName;
    private String accHolderName;
    private long accNumber;

    public String getBankName() {
        return bankName;
    }
    public void setBankName(String bankName) {
        this.bankName = bankName;
    }
    public String getAccHolderName() {
        return accHolderName;
    }
    public void setAccHolderName(String accHolderName) {
        this.accHolderName = accHolderName;
    }
    public long getAccNumber() {
        return accNumber;
    }
    public void setAccNumber(long accNumber) {
        this.accNumber = accNumber;
    }
}// End of the BankDetails class.
```

Step 3

Create a **BankCustomer** class (Adapter class).

File: BankCustomer.java

```
// This is the adapter class

import java.io.BufferedReader;
import java.io.InputStreamReader;
public class BankCustomer extends BankDetails implements CreditCard {
 public void giveBankDetails(){
  try{
   BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

   System.out.print("Enter the account holder name :");
   String customername=br.readLine();
   System.out.print("\n");

   System.out.print("Enter the account number:");
   long accno=Long.parseLong(br.readLine());
   System.out.print("\n");

   System.out.print("Enter the bank name :");
   String bankname=br.readLine();
  setAccHolderName(customername);
  setAccNumber(accno);
  setBankName(bankname);
  }catch(Exception e){
       e.printStackTrace();
  }
 }
 @Override
 public String getCreditCard() {
  long accno=getAccNumber();
  String accholdername=getAccHolderName();
  String bname=getBankName();

  return ("The Account number "+accno+" of "+accholdername+" in "+bname+ "
                      bank is valid and authenticated for issuing the credit card. ");
 }
}//End of the BankCustomer class.
```

Step 4

Create a **AdapterPatternDemo** class (client class).

File: AdapterPatternDemo.java

```
//This is the client class.
public class AdapterPatternDemo {
 public static void main(String args[]){
  CreditCard targetInterface=new BankCustomer();
  targetInterface.giveBankDetails();
  System.out.print(targetInterface.getCreditCard());
 }
}//End of the BankCustomer class.
```

download this example

Output

```
Enter the account holder name :Sonoo Jaiswal

Enter the account number:10001

Enter the bank name :State Bank of India

The Account number 10001 of Sonoo Jaiswal in State Bank of India bank is valid
and authenticated for issuing the credit card.
```

← prev next →

Please Share

Learn Latest Tutorials

Testing

NumPy

Verbal A.

AWS

D. Math.

React Native

Preparation

Aptitude

Reasoning

Verbal A.

Interview

B.Tech / MCA

DBMS

DS

DAA

OS

C. Network

Compiler D.

COA

Web Tech.

Cyber Sec.	C	C++	Java
.Net	Python	Programs	Control S.