Le patron "Adapter"

Le patron adapter : Les adaptateurs (1/7)

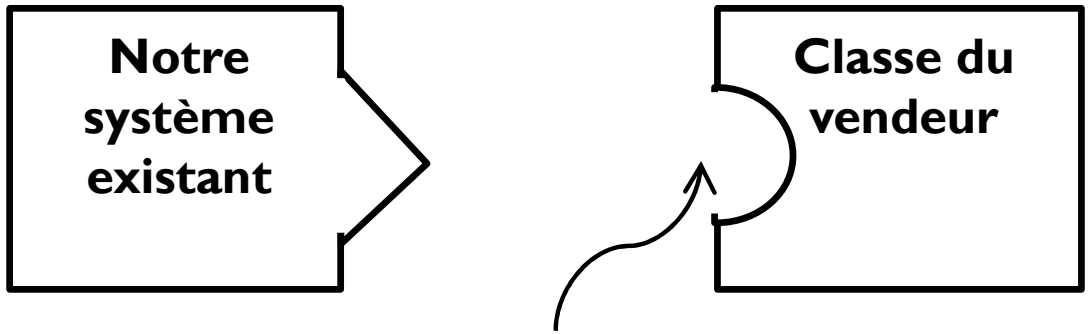
Prise Européenne

Adaptateur

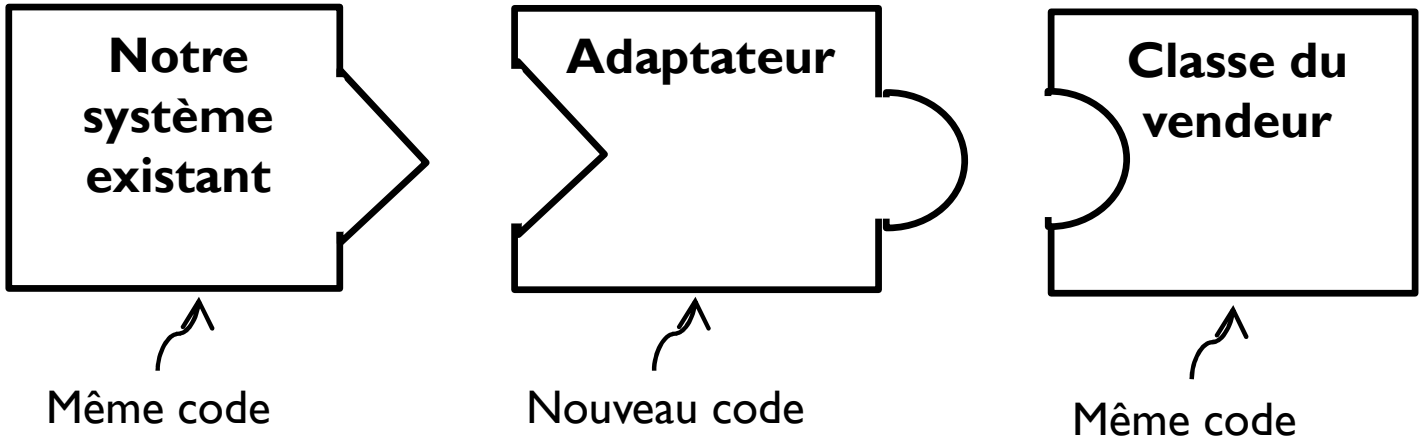
Câble American du PC

L'adaptateur convertit une interface à une autre

Le patron adapter : Les adaptateurs dans l'OO (2/7)



L'interface ne correspond par à ce qu'on a déjà codé.
Ça ne va pas marché!
(supposant qu'on ne peut pas changer de vendeur)



Le patron adapter : Dinde et Canard (3/7)

- Supposant que le dinde marche et cancane comme le canard

Un canard peut cancaner et voler

```
public interface Duck{
void quack();
void fly();
}
```

Une simple implémentation du comportement du canard

```
public class MallardDuck implements Duck{
    public void quack(){
    System.out.println("Quack");}
    public void fly(){
    System.out.println("Fly");}
}
```

```
public interface Turkey{
void gobble();
void fly();
}
```

Le dinde ne cancane pas, mais glougloute

Le dinde peut voler (courte distance)

```
public class WildTurkey implements Turkey{
    public void gobble(){
    System.out.println("Gobble");}
    public void fly(){
    System.out.println("Fly for a short distance");}
}
```

Le patron adapter : L'adaptateur du dinde(4/7)

- Supposons qu'on a un manque de canards et on va utiliser des dindes à leur pace ➔ Il faut écrire un "adapter"

Respecter l'interface des canards

```
public class TurkeyAdapter implements Duck{
private Turkey turkey;
public TurkeyAdapter(Turkey turkey){
this.turkey=turkey;}
@Override
public void quack(){
   turkey.gobble();
}
@Override
public void fly(){
   for(int i=0;i<5;i++)
       {turkey.fly();}
}
}
```

Une référence vers l'objet à adapter

Translation des méthodes

Le patron adapter : Testons l'adaptateur(5/7)

```
public class TestAdapter{
public static void main (String[] arg)
{
    Duck mallard= new MallardDuck();
    Turkey wild = new WildTurkey();
    Duck turkeyAdapter = new TurkeyAdapter(wild);
    Duck[] tab = new Duck[2];
    tab[0] = mallard;
    tab[1] = turkeyAdapter ;
    for(int i =0; i<2;i++)
    {
            tab[i].quack();
            tab[i].fly();
    }

}
}
```

- Donner le résultat d'exécution de cette classe

Le patron adapter (6/7)

- Définition: **Adapter**
  - Le **patron Adapter** convertit l'interface d'une classe à une autre interface que le client attend. Les adaptateurs permettent aux classes, aillant des interfaces incompatibles, de travailler ensemble.

Adapter :
Le diagramme de classes du patron (7/7)