Adapter Design Pattern

Daniel Muresan

30432

adapt verb

\ ə-ˈdapt, a-\

adapted; **adapting**; **adapts**

Definition of *adapt*

transitive verb

: to make fit (as for a new use) often by modification
// *adapt* the curriculum to students' needs

intransitive verb

: to become adapted
// *adapt* to a new enviroment

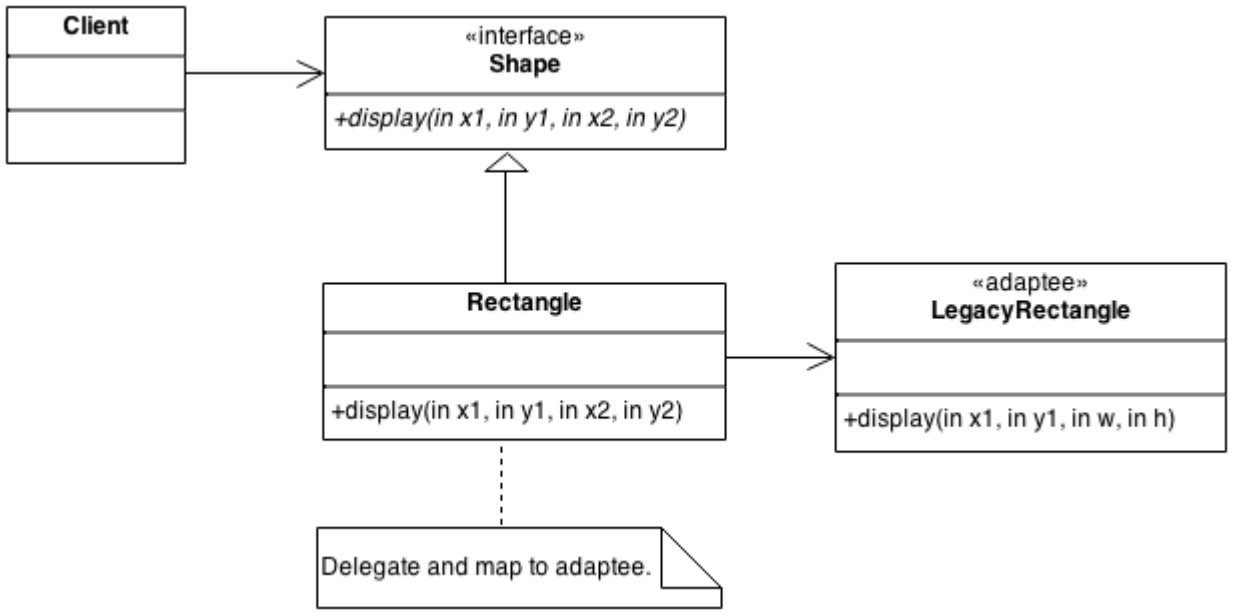
Intent

- 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

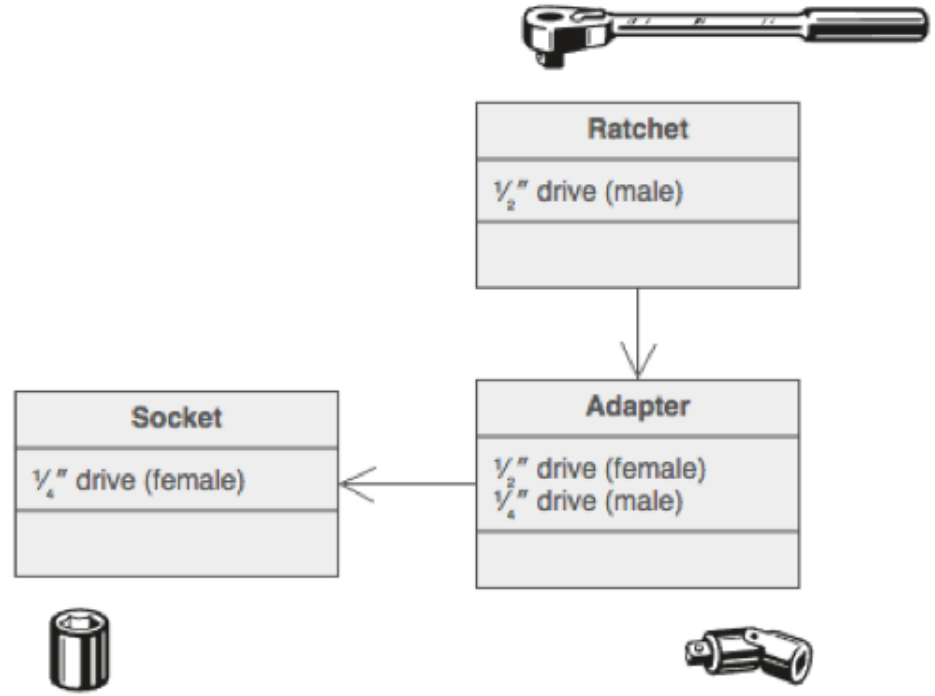
Problem

An "off the shelf" component offers compelling functionality that you would like to reuse, but its "view of the world" is not compatible with the philosophy and architecture of the system currently being developed.

Structure


Client
«interface»
Shape
+display(in x1, in y1, in x2, in y2)
Rectangle
+display(in x1, in y1, in x2, in y2)
«adaptee»
LegacyRectangle
+display(in x1, in y1, in w, in h)
Delegate and map to adaptee.

Example


Ratchet
½" drive (male)
Socket
¼" drive (female)
Adapter
½" drive (female)
¼" drive (male)

Code

```
interface Shape {
    void display(int x, int y, int z, int j);
}


class LegacyRectangle {
    public void display(int x, int y, int width, int height) {
    System.out.println("Rectangle with coordinate left-down point (" + x + ";" + y + "), width: " + width
    + ", height: " + height);
    }

}

class Rectangle implements Shape {
    private LegacyRectangle adaptee;

    public Rectangle(LegacyRectangle rectangle) {
        this.adaptee = rectangle;
    }

    @Override
    public void display(int x1, int y1, int x2, int y2) {
        int x = Math.min(x1, x2);
        int y = Math.min(y1, y2);
        int width = Math.abs(x2 - x1);
        int height = Math.abs(y2 - y1);
        adaptee.display(x, y, width, height);
    }
}
```

Advantages:

- Helps achieve reusability and flexibility.
- Client class is not complicated by having to use a different interface and can use polymorphism to swap between different implementations of adapters.

Disadvantages:

- All requests are forwarded, so there is a slight increase in the overhead.
- Sometimes many adaptations are required along an adapter chain to reach the type which is required.