Visitor
Visitor Design Pattern

Group 09
Võ Trần Thanh Lương
Tô Minh Thành
Nguyễn Trần Trọng Tâm

Contents

Problems and normal solutions

Problem

Normal solution

Solution:

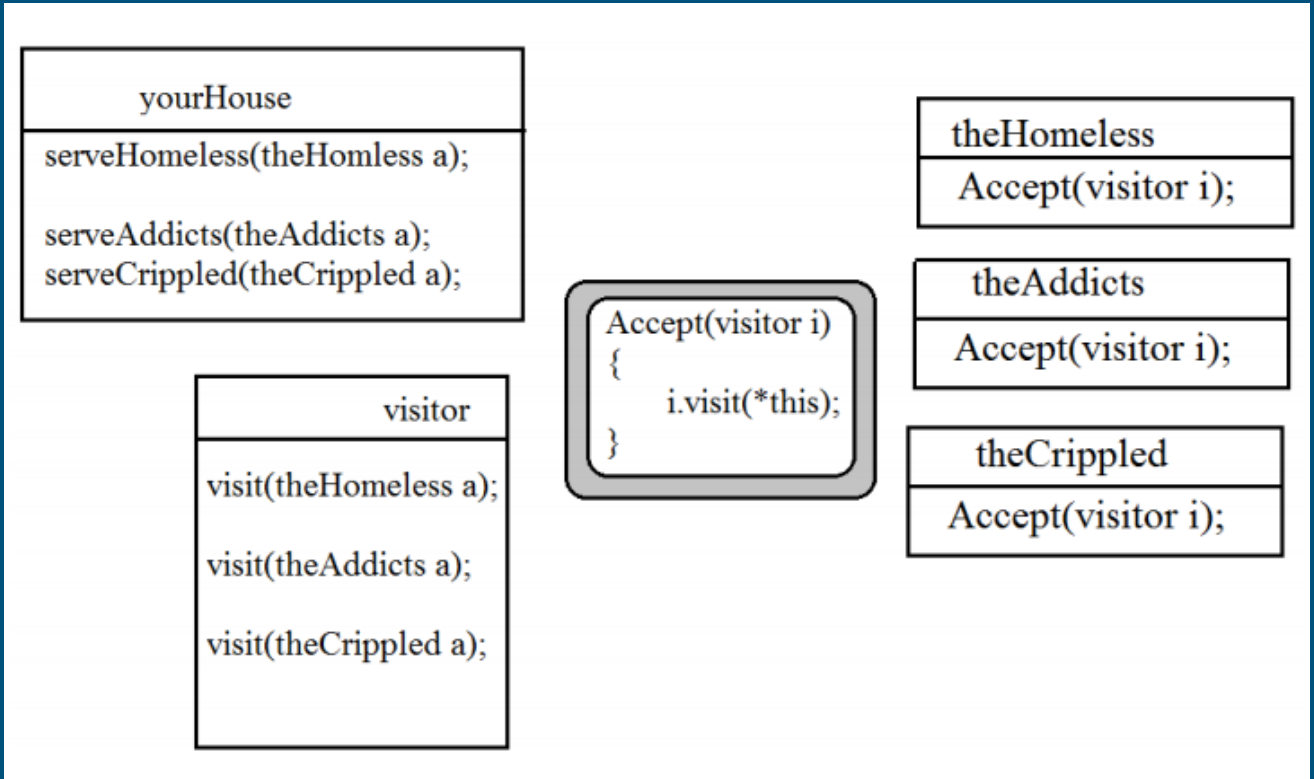
Trying to modify the predefined structure when a new feature is needed.

Problem with this solution:

Expensive and inefficient since modifications rare required constantly.

Solving the problem using pattern

Look again at the problem



Visitor Design Pattern

Visitor Design Pattern

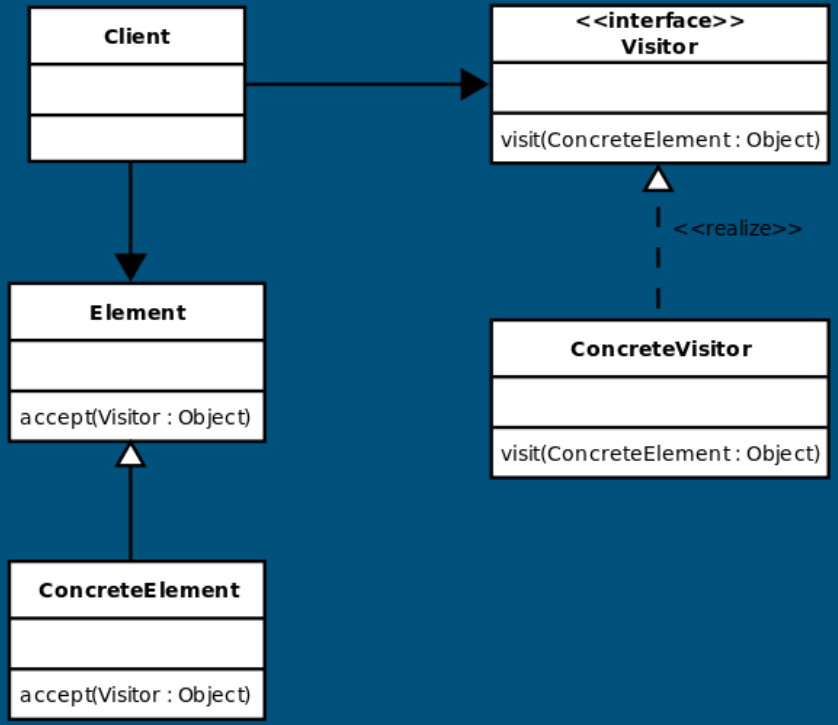
Definition (by the Gang of Four):

“A way of separating an operation from an object structure on which it operates…”

Main purpose:

Add new methods to a class without having to modify the source.

Structure


Client
<<interface>>
Visitor
visit(ConcreteElement : Object)
<<realize>>
Element
accept(Visitor : Object)
ConcreteVisitor
visit(ConcreteElement : Object)
ConcreteElement
accept(Visitor : Object)

Similar problems

Most of OOP languages support **single dispatch**, or **virtual method**

```
class SpaceShip {
  virtual string GetShipType() {
    return "SpaceShip";
  }
}

class ApolloSpacecraft : SpaceShip {
  virtual string GetShipType() {
    return "ApolloSpacecraft";
  }
}
```

—

```
Int main(){
SpaceShip ship = new ApolloSpacecraft();
cout<<ship.GetShipType()<<endl;
}
```

Ouput on the screen would be

ApolloSpacescraft;

But suppose you have something like this :

```
class Asteroid {
 virtual void CollideWith(SpaceShip ship) {
  cout<<"Asteroid hit a SpaceShip";
  }
  virtual void CollideWith(ApolloSpacecraft ship) {
   cout<<"Asteroid hit an ApolloSpacecraft");
  }
};

class ExplodingAsteroid : Asteroid {
 void CollideWith(SpaceShip ship) {
   cout<<"ExplodingAsteroid hit a SpaceShip";
  }
 void CollideWith(ApolloSpacecraft ship) {
    cout<<"ExplodingAsteroid hit an ApolloSpacecraft";
  }
};
```

```
Int main()
{

Asteroid theExplodingAsteroidRef = new
ExplodingAsteroid();

SpaceShip theApolloSpacecraftRef = new
ApolloSpacecraft();

theExplodingAsteroidRef.CollideWith(theApolloSp
acecraftRef);
}
```

ExplodingAsteroid hit a SpaceShip

Pros and Cons

Pros

1) Allow additional operations to be operated on structure without having to modify the structure itself.

2) Allow the result of implementation to depend on both the the return data type and the parameter type.

Cons

1) Additional implementation is required.

2) If visitor pattern was not intended, modification is required.

THANK YOU FOR YOUR ATTENTION

References

- https://sourcemaking.com/design_patterns/visitor
- https://www.codeproject.com/articles/588882/the-visitor-pattern-explained