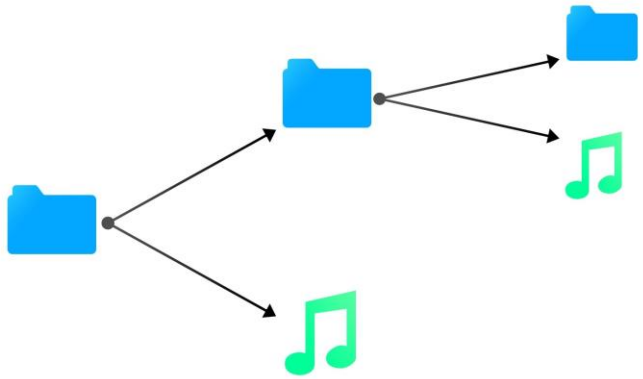
组合(Composite)模式

波波老师~研发总监/资深架构师

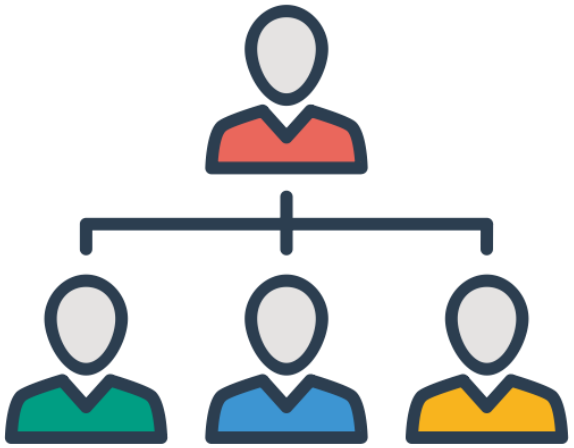
定义

- 将对象组织成**树形结构**，以表示整体部分关系。
- 让客户能够统一方式处理单个和组合对象
- 实现利用了递归

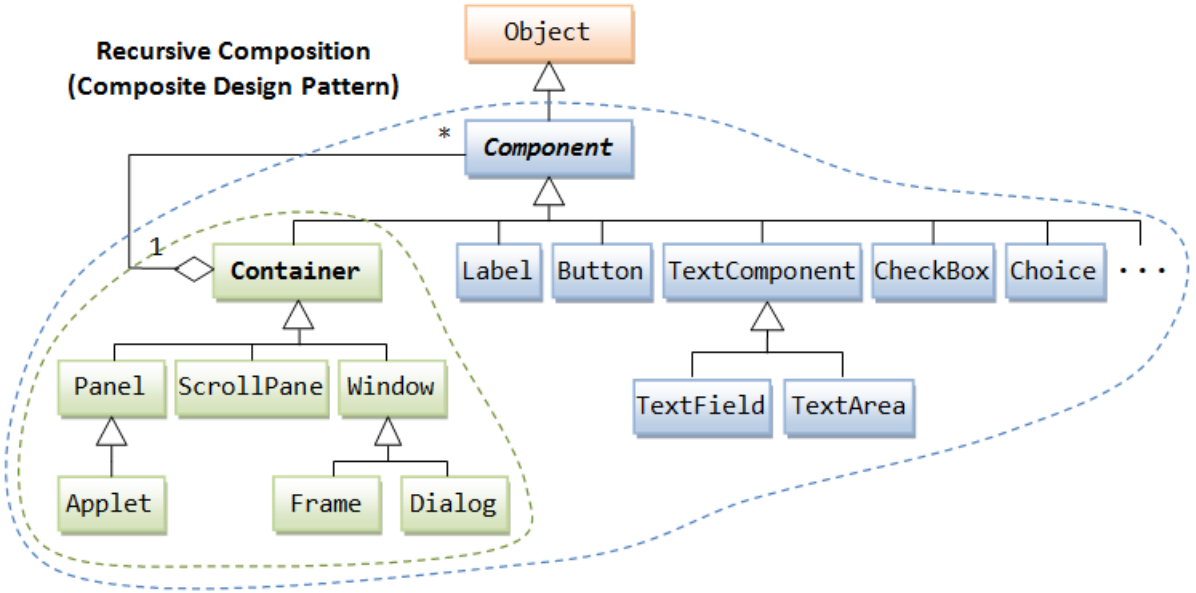
树形结构



目录结构

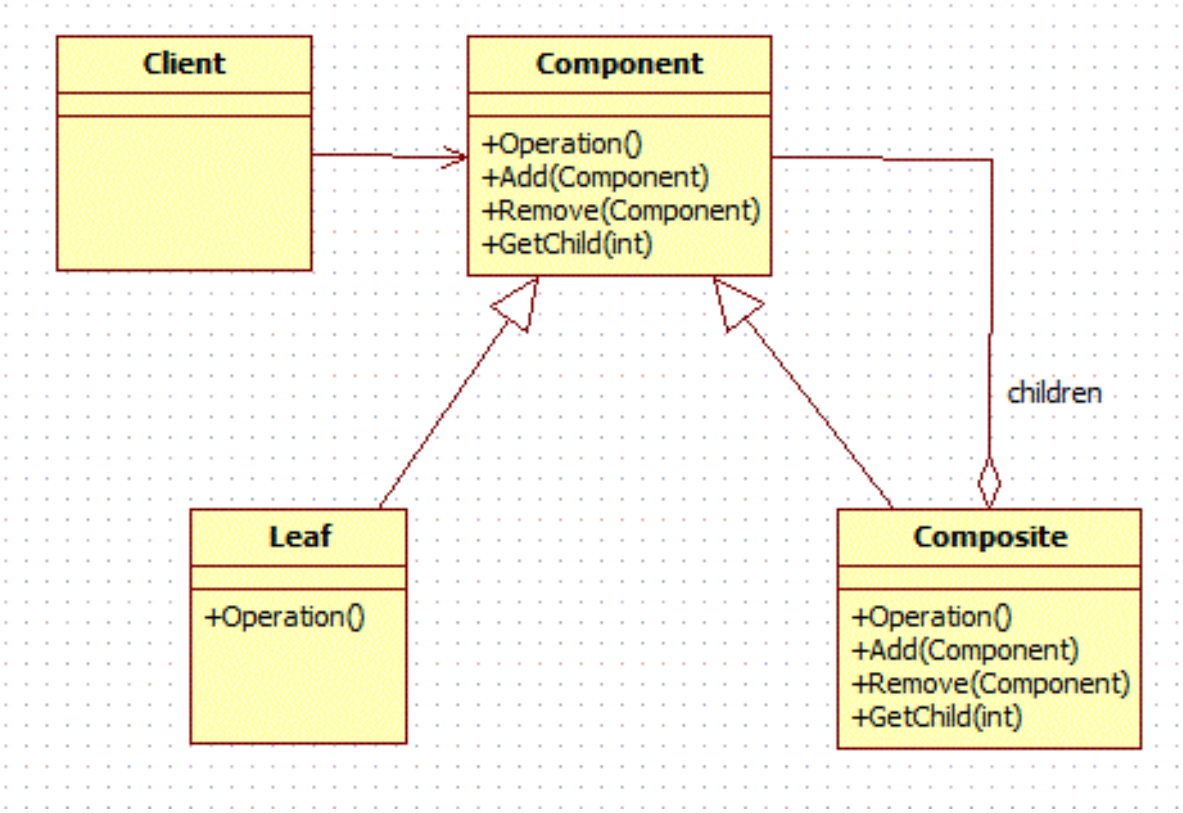


组织结构



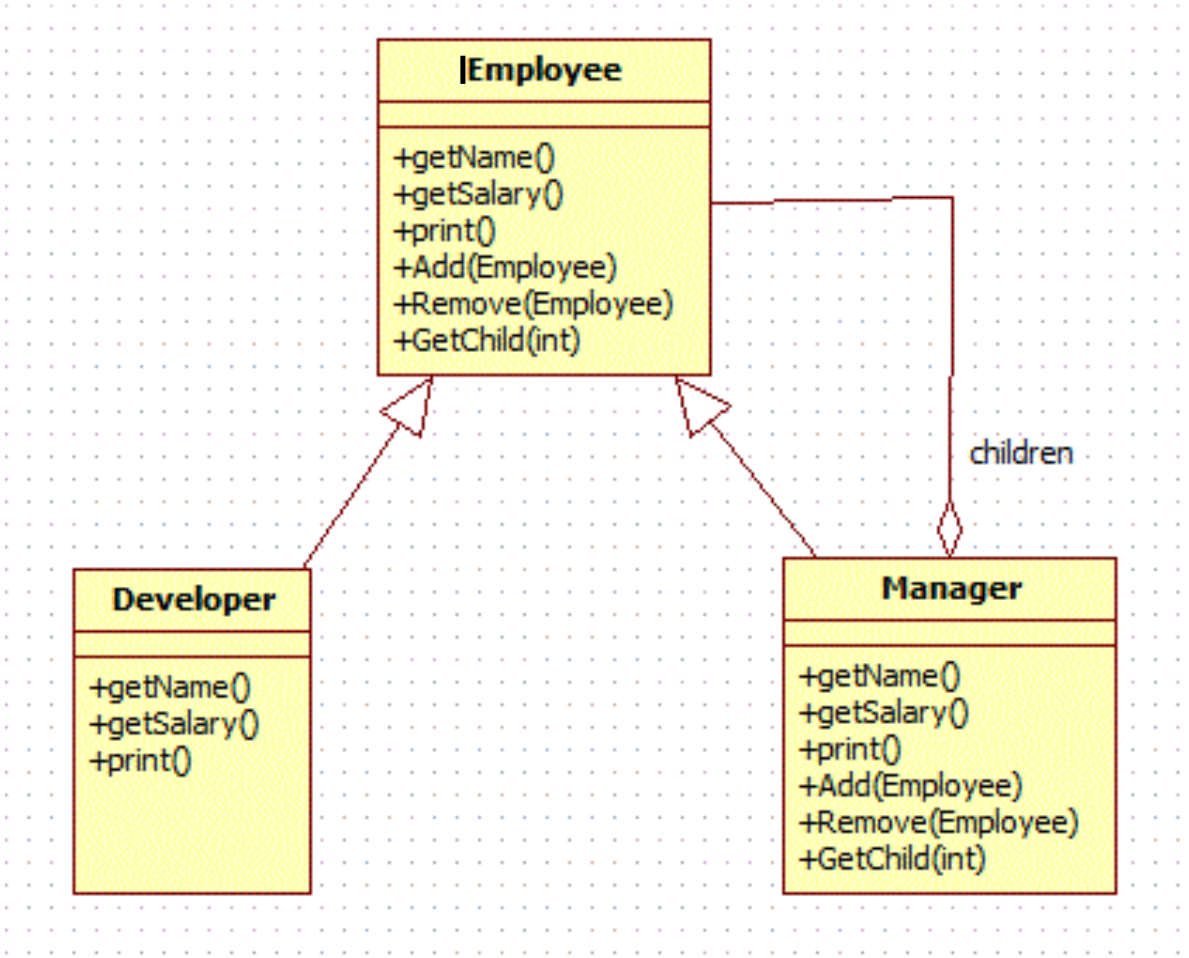
GUI组件结构

关系图



角色	职责
Component	组合中对象的公共接口， 也可以实现组合/叶子的公共行为 也定义访问和管理子部件的接口
Leaf	表示组合中的叶子对象。叶子没有孩子节点 定义原子操作
Composite	定义具有孩子节点的对象行为 存储孩子节点，并实现对孩子节点的操作
Client	通过Component接口操作组合中的对象

案例~打印组织雇员薪水


Employee
+getName()
+getSalary()
+print()
+Add(Employee)
+Remove(Employee)
+GetChild(int)
children
Developer
+getName()
+getSalary()
+print()
Manager
+getName()
+getSalary()
+print()
+Add(Employee)
+Remove(Employee)
+GetChild(int)

实现~Component接口

```java
package io.spring2go.corespring.composite_simple;

// Component
public interface IEmployee {
    public void add(IEmployee employee);
    public void remove(IEmployee employee);
    public IEmployee getChild(int i);
    public String getName();
    public double getSalary();
    public void print();
}
```

实现~Leaf节点

```java
// leaf
public class Developer implements IEmployee {

    private String name;
    private double salary;

    public Developer(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    @Override
    public void add(IEmployee employee) {
        String msg = "not supported by leaf node.";
        throw new UnsupportedOperationException(msg);
    }

    @Override
    public void remove(IEmployee employee) {
        String msg = "not supported by leaf node.";
        throw new UnsupportedOperationException(msg);
    }

    @Override
    public IEmployee getChild(int i) {
        String msg = "not supported by leaf node.";;
        throw new UnsupportedOperationException(msg);
    }

    @Override
    public String getName() {
        return this.name;
    }

    @Override
    public double getSalary() {
        return this.salary;
    }

    @Override
    public void print() {
        System.out.println("-------------");
        System.out.println("Name ="+getName());
        System.out.println("Salary ="+getSalary());
        System.out.println("-------------");
    }
}
```

实现~Composite节点

```java
// Composite
public class Manager implements IEmployee {

    private String name;
    private double salary;
    private List<IEmployee> employees = new ArrayList<IEmployee>();

    public Manager(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    @Override
    public void add(IEmployee employee) {
        employees.add(employee);
    }

    @Override
    public void remove(IEmployee employee) {
        employees.remove(employee);
    }

    @Override
    public IEmployee getChild(int i) {
        return employees.get(i);
    }

    @Override
    public String getName() {
        return this.name;
    }

    @Override
    public double getSalary() {
        return this.salary;
    }

    @Override
    public void print() {
        System.out.println("-------------");
        System.out.println("Name =" + getName());
        System.out.println("Salary =" + getSalary());
        System.out.println("-------------");

        for(IEmployee employee : this.employees) {
            employee.print();
        }
    }
}
```

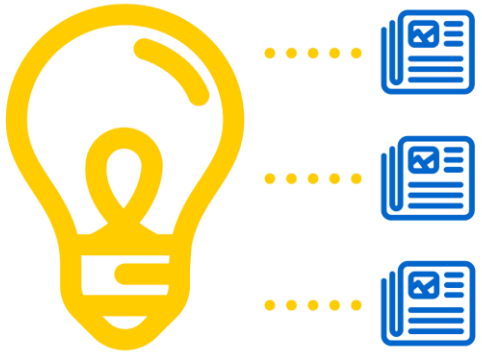
实现~Client端

```java
package io.spring2go.corespring.composite_simple;

// Client
public class CompositePatternMain {

    public static void main(String[] args) {
        IEmployee emp1 = new Developer("John", 10000);
        IEmployee emp2 = new Developer("David", 15000);
        IEmployee manager1 = new Manager("Daniel", 25000);
        manager1.add(emp1);
        manager1.add(emp2);
        IEmployee emp3 = new Developer("Michael", 20000);
        Manager generalManager = new Manager("Mark", 50000);
        generalManager.add(emp3);
        generalManager.add(manager1);
        generalManager.print();
    }

}
```

```
-------------
Name =Mark
Salary =50000.0
-------------
-------------
Name =Michael
Salary =20000.0
-------------
-------------
Name =Daniel
Salary =25000.0
-------------
-------------
Name =John
Salary =10000.0
-------------
-------------
Name =David
Salary =15000.0
-------------
```

应用

- java.awt.Container#add(Component)
- javax.faces.component.UIComponent#getChildren()



参考

- Composite design pattern in java
  - https://java2blog.com/composite-design-pattern-in-java/
- An Absolute Beginner’ s Tutorial for Understanding and Implementing Composite Pattern in C#
  - https://www.codeproject.com/Articles/1197606/An-Absolute-Beginner-s-Tutorial-for-Understanding

代码

- https://github.com/spring2go/core-spring-patterns

波波微课

spring2go.com