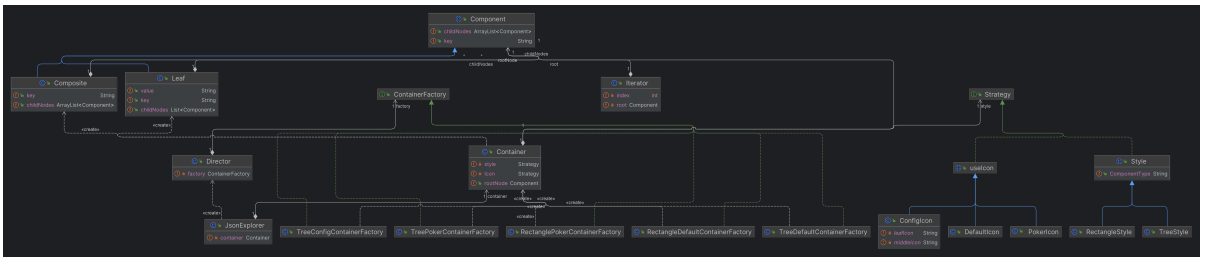
FJE进阶

实验要求

对已有FJE实现进行设计重构，改用迭代器+策略者模式

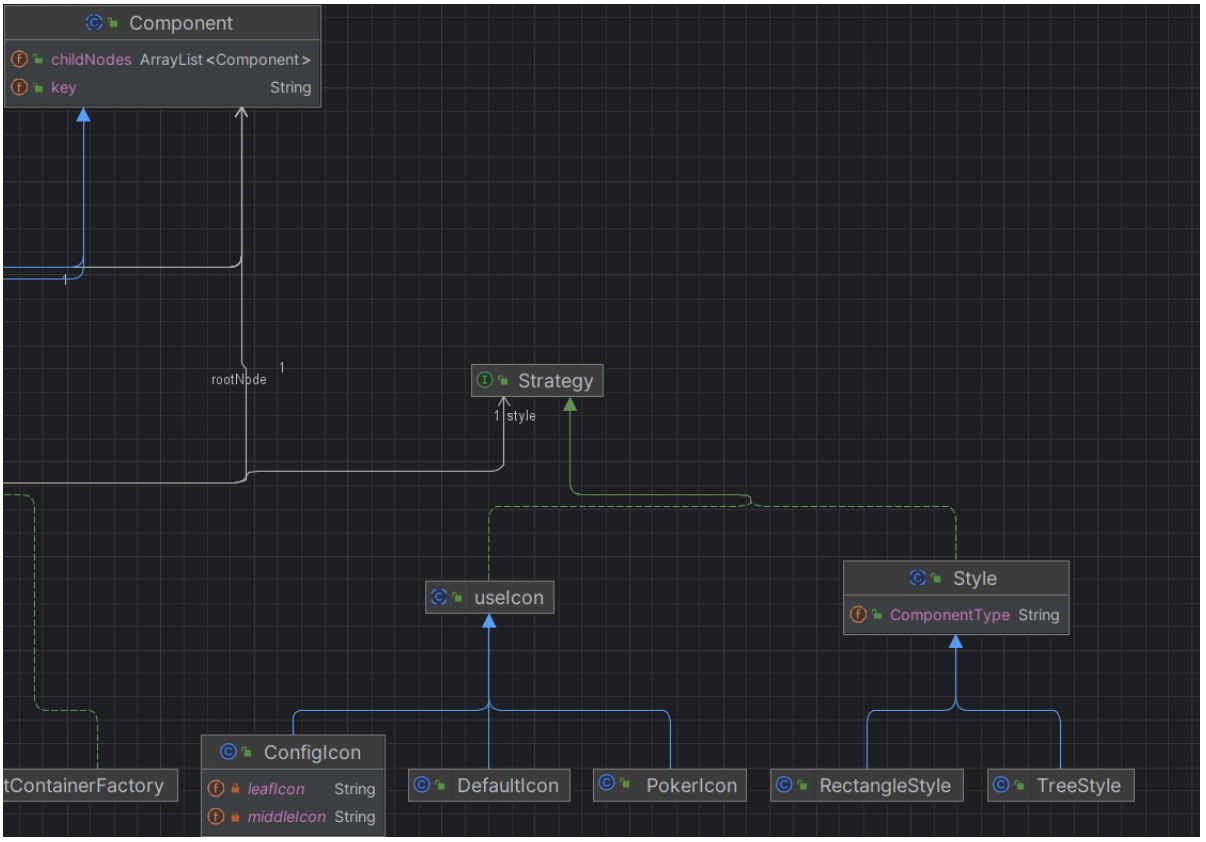
设计方法

UML类图



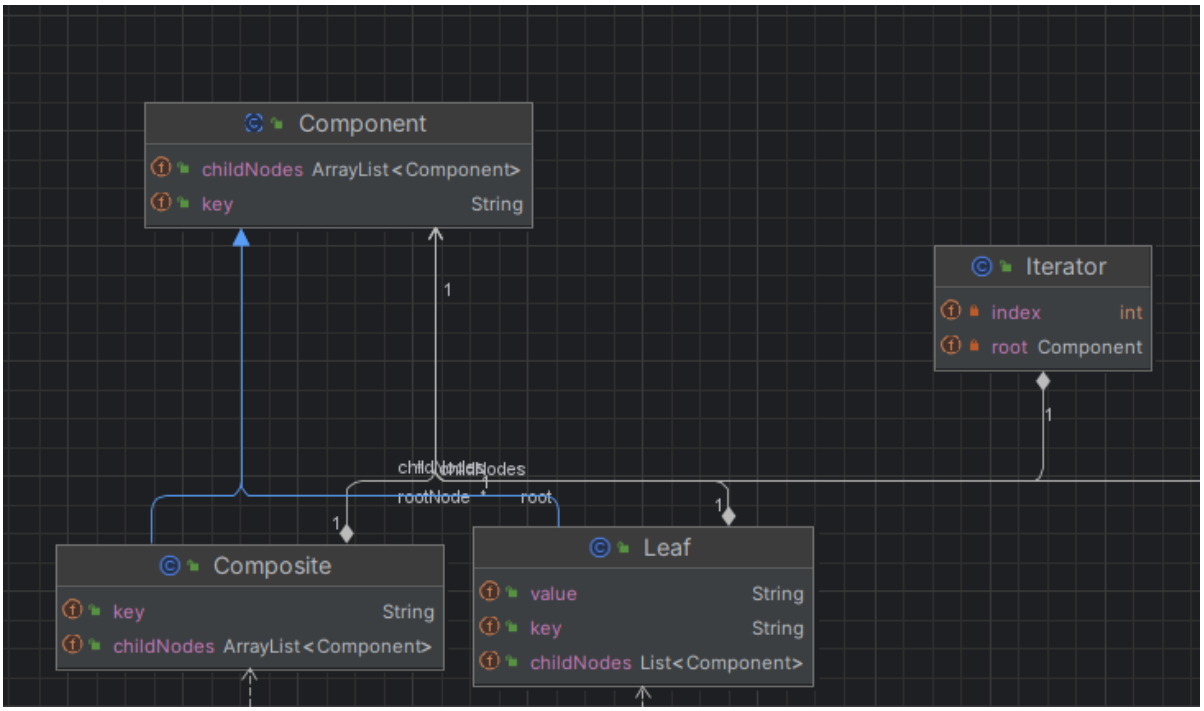
迭代器：

`Iterator`类和`Component`类及其子类（`Composite`和`Leaf`）体现了迭代器模式，通过`Iterator`遍历`Component`的实例,而不暴露该对象的内部表示。



策略模式：

`Strategy`接口及其具体实现类和`Container`类体现了策略模式，通过在`Container`中使用不同的`Style`以及`useIcon`策略实现动态变化。



具体实现

`Iterator`类

```
public class Iterator {
    // 当前遍历的位置索引
    private int index;
    // 要遍历的根组件
    private Component root;

    // 构造函数，接收一个根组件，并将索引初始化为0
    Iterator(Component root){
        this.root = root;
        index = 0;
    }

    // 获取下一个节点，如果没有子节点或者遍历完毕，则返回null
    public Component getNextNode(){
        // 如果根节点没有子节点，返回null
        if(root.childNodes == null) return null;

        // 如果当前索引小于子节点的数量，返回当前索引位置的子节点，并将索引加1
        if(index < root.childNodes.size()){
            return root.childNodes.get(index++);
        }

        // 如果已经遍历完所有子节点，返回null
        return null;
    }

    // 检查是否已经遍历完所有子节点
    public Boolean isEnd(){
        // 当索引等于子节点数量时，表示遍历结束
        return index == root.childNodes.size();
    }
}
```

`Strategy`类

```
public interface Strategy {
    public String getType();
    public abstract void draw(Component n, Strategy useIcon);
    public abstract String getLeafIcon();
    public abstract String getMiddleIcon();
}
```

实验结果