单例模式实现代码

```
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;

public class Single {
    private static Single single;
    private static Lock lock = new ReentrantLock();

    private Single(){}

    public static Single GetInstance() {
        if (single != null) {
            return single;
        }
        lock.lock();
        if (single != null) {
            return single;
        }
        single = new Single();
        lock.unlock();
        return single;
    }
}
```

组合模式应用

```
package form;

public class Main {
    public static void main(String[] args) {

        Frame frame = new Frame();
        Window window = new Window("窗口1");
        frame.addChild(window);

        Panel p = new Panel("面板1");
        window.addChild(p);

        Button button = new Button("按钮1");
        CheckBox checkBox = new CheckBox("复选框1");
        p.addChild(button);
        p.addChild(checkBox);

        Window window2 = new Window("窗口2");
        frame.addChild(window2);

        Panel p2 = new Panel("面板2");
        window2.addChild(p2);

        Button button2 = new Button("按钮2");
        CheckBox checkBox2 = new CheckBox("复选框2");
```

```
        TextField  textField = new TextField("文本框");
        p2.addChild(button2);
        p2.addChild(checkBox2);
        p2.addChild(textField);

        UIContext  context =  UIContext.withContext(null, frame);

        frame.draw(context);
    }
}
package form;

public abstract class Componet {
    private int deep;
    private String name;

    abstract void draw(UIContext context);
    public int getDeep() {
        return deep;
    }

    public void setDeep(int deep) {
        this.deep = deep;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
package form;

public class Button extends Componet {

    public Button(String name) {
        setName(name);
    }
    public void draw(UIContext context) {
        int deep = getDeep();
        for (int i = 0; i < deep; i++) {
            System.out.print(" ");
        }
        System.out.println("draw Button:" + getName());
    }
}
package form;

public class CheckBox extends Componet {
    public CheckBox(String name) {
        setName(name);
    }

    public void draw(UIContext context) {
        int deep = getDeep();
        for (int i = 0; i < deep; i++) {
```

```
            System.out.print(" ");
        }
        System.out.println("draw CheckBox:" + getName());
    }
}
package form;

import java.util.ArrayList;
import java.util.List;

public class Container extends Componet {
    private final List<Componet> componetList = new ArrayList<Componet>();

    public void draw(UIContext context) {
        UIContext uc = UIContext.withContext(context, this);
        for (Componet c : componetList) {
            c.draw(uc);
        }
    }

    public void addChild(Componet componet) {
        if (null != componet) {
            componet.setDeep(this.getDeep() + 1);
            componetList.add(componet);
        }
    }
}
package form;

public class Frame extends Container {
    @Override
    public void draw(UIContext context) {
        int deep = getDeep();
        for (int i = 0; i < deep; i++) {
            System.out.print(" ");
        }
        System.out.println("draw Frame");
        super.draw(context);
    }
}
package form;

public class Panel extends Container {
    public Panel(String name) {
        setName(name);
    }

    @Override
    public void draw(UIContext context) {
        int deep = getDeep();
        for (int i = 0; i < deep; i++) {
            System.out.print(" ");
        }
        System.out.println("draw Panel:" + getName());
        super.draw(context);
    }
}
package form;
```

```
public class TextField extends Componet {
    public TextField(String name) {
        setName(name);
    }

    @Override
    void draw(UIContext context) {
        int deep = getDeep();
        for (int i = 0; i < deep; i++) {
            System.out.print(" ");
        }
        System.out.println("draw TextField:" + getName());
    }
}
package form;

public class UIContext {
    private Componet componet;
    private UIContext parentContext;

    public static UIContext withContext(UIContext context, Componet componet) {
        UIContext ctx = new UIContext();
        ctx.componet = componet;
        ctx.parentContext = context;
        return ctx;
    }

    public Componet getParent() {
        return componet;
    }
}
package form;

public class UIContext {
    private Componet componet;
    private UIContext parentContext;

    public static UIContext withContext(UIContext context, Componet componet) {
        UIContext ctx = new UIContext();
        ctx.componet = componet;
        ctx.parentContext = context;
        return ctx;
    }

    public Componet getParent() {
        return componet;
    }
}
```

结果输出

```
draw Frame
 draw Window:窗口1
  draw Panel:面板1
   draw Button:按钮1
   draw CheckBox:复选框1
 draw Window:窗口2
  draw Panel:面板2
   draw Button:按钮2
   draw CheckBox:复选框2
   draw TextField:文本框
```