KEYHOLE SOFTWARE | Educational Offering

Design Pattern Primer

Exercise: Memento

Overview

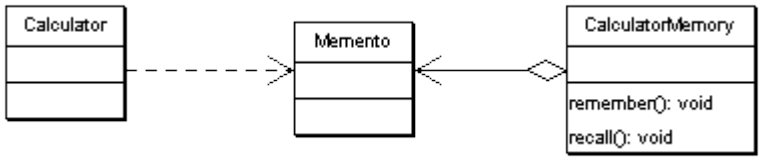
In this exercise, you will apply the Memento pattern to the calculator implementation.

User Requirement

You need to utilize Java compiler and byte code interpreter to create and compile classes, methods, and fields.

Introduction

The Memento pattern allows encapsulated state of an object to be saved and restored when necessary. This exercise applies this pattern to implement calculation result memory. Since the patterns design calls for a Caretaker object that requests and holds on to *Memento* instances. A *CalculatorMemory* class will play the role of the patterns *Caretaker* role, and provide methods to remember and recall calculator results.



Source for this exercise is defined in the **db.lab.memento** package.

Exercise Instructions

1. Produce Memento

In the ***db.lab.memento*** package add the inner class and method definitions shown below to the Calculator class. The inner class definition allows access to private state of the enclosing class. This is key to the *Memento* pattern ability to access encapsulated state of its subject.

```
        Memento mem = new Memento();
        mem.result = this.result;

        return mem;
    }
```

```
	public void applyMemento(Object o) {

		result = ((Memento) o).result;

	}

	// memento inner class
	public class Memento {

		double result = 0.00;
		String operation = null;

		public void remember(Calculator calc) {

			result = getResult();
			operation = calc.getOperation();
		}

		public void recall(Calculator calc) {

			calc.setResult(result);

		}
```

2. Implement the CalculatorMemory Class

The memento pattern introduces a *Caretaker* class that is responsible for obtaining a *Memento* instance from an *Originator*(*Calculator*) and then upon request restoring state to a specific *Memento* instance. The Caretaker role is carried out by the *CalculatorHistory* class. It's implementation is shown below. Implement this class in the *dp.lab.memento* package. Study how the implementation remembers and recalls a Calculator result.

```
import java.util.Vector;

public class CalculatorMemory {

	Vector mementos = new Vector();

	public void remember(Calculator calc) {

		mementos.addElement(calc.produceMemento());
	}

	public void recall(Calculator calc) {

		if (!mementos.isEmpty())
		{
			Object o = mementos.elementAt(0);
			calc.applyMemento(o);
			mementos.removeElementAt(0);
		}

	}
```

```
}
```

3. Test the implementation
Implement and execute the Tester class shown below. Study the implementation.

```
public class Tester {

	public static void main(String[] args) {


		// basic calculator

		System.out.println("* * Basic Calculator * *");

		Calculator calc = PrototypeFactory.basic();

		// Create Memory caretaker

		CalculatorMemory memory = new CalculatorMemory();

		calc.execute("+", 10.0);
		// remember result
		memory.remember(calc);
		calc.execute("/", 2.0);
		// recall from memory
		memory.recall(calc);
		calc.execute("*",5.0);
		calc.print();


	}

}
```