State Pattern(Source Code)

the InitState, S0, S1, S2, S3, S4, S5, S6, State, StateMachine classes are responsible for the state pattern.

sorce code in State Class:

```
public abstract class State {
    StateMachine model;

    public State(StateMachine model) {
        this.model = model;
    }

    void activate(){
        System.out.println("error message: can not impletment this operation in this
state!");
    }

    void start(){
        System.out.println("error message: can not impletment this operation in this
state");
    }

    /*
        credit:      t=1
        cash:        t=2
     */
    void payType(int t){
        System.out.println("error message: can not impletment this operation in this
state");
    }

    void approved(){
        System.out.println("error message: can not impletment this operation in this
state");
    }

    void reject(){
        System.out.println("error message: can not impletment this operation in this
state");
    }

    /*
       Regular:    g=1
       Super:      g=2
       Premium:    g=3
     */
    void selectGas(int g){
        System.out.println("error message: can not impletment this operation in this
state");
    }

    void cancel(){
        System.out.println("error message: can not impletment this operation in this
state");
```

```
    }

    void startPump(){
        System.out.println("error message: can not impletment this operation in this 
state");
    }

    void pump(){
        System.out.println("error message: can not impletment this operation in this 
state");
    }

    void stopPump(){
        System.out.println("error message: can not impletment this operation in this 
state");
    }

    void receipt(){
        System.out.println("error message: can not impletment this operation in this 
state");
    }

    void noReceipt(){
        System.out.println("error message: can not impletment this operation in this 
state");
    }
```

sorce code in InitState Class:

```
class InitState extends State {

    InitState(StateMachine model) {
        super(model);
    }

    @Override
    void activate() {
        if (model.s == model.LS[7]) {
            model.s = model.LS[0];
            model.getOP().StoreData();
        }
    }
}
```

sorce code in S0 Class:

```
class S0 extends State {

    S0(StateMachine model) {
        super(model);
    }

    @Override
    void start() {
        if (model.s == model.LS[0]) {
            model.s = model.LS[1];
            model.getOP().PayMsg();
        }
    }
}
```

sorce code in S1 Class:

```
class S1 extends State {

    S1(StateMachine model) {
        super(model);
    }

    /*
        credit: t=1
        cash:   t=2
     */
    @Override
    void payType(int t) {
        if ((t == 1) && (model.s == model.LS[1])) {
            model.s = model.LS[2];
        } else if ((t == 2) && (model.s == model.LS[1])) {
            model.s = model.LS[3];
            model.getOP().StoreCash();
            model.getOP().DisplayMenu();
        }
    }
}
```

sorce code in S2 Class:

```
class S2 extends State {

    S2(StateMachine model) {
        super(model);
    }

    @Override
    void approved() {
        if (model.s == model.LS[2]) {
            model.s = model.LS[3];
            model.getOP().DisplayMenu();
        }
    }

    @Override
    void reject() {
        if (model.s == model.LS[2]) {
            model.s = model.LS[0];
            model.getOP().RejectMsg();
        }
    }
}
```

sorce code in S3 Class:

```
class S3 extends State {

    S3(StateMachine model) {
        super(model);
    }
    @Override
    void selectGas(int g) {
        if (model.s == model.LS[3]) {
            model.s = model.LS[4];
            model.getOP().SetPrice(g);
        }
    }

    @Override
    void cancel() {
        if (model.s == model.LS[3]) {
            model.s = model.LS[0];
            model.getOP().CancelMsg();
            model.getOP().ReturnCash();
        }
    }
}
```

sorce code in S4 Class:

```
class S4 extends State {

    S4(StateMachine model) {
        super(model);
    }

    @Override
    void startPump() {
        if (model.s == model.LS[4]) {
            model.s = model.LS[5];
            model.getOP().SetInitialValues();
            model.getOP().ReadyMsg();
        }
    }
}
```

sorce code in S5 Class:

```
class S5 extends State {

    S5(StateMachine model) {
        super(model);
    }

    @Override
    void pump() {
        if (model.s == model.LS[5]) {
            // stay in the same state
            model.getOP().PumpGasUnit();
            model.getOP().GasPumpedMsg();
        }
    }

    @Override
    void stopPump() {
        if (model.s == model.LS[5]) {
            model.s = model.LS[6];
            model.getOP().StopMsg();
        }
    }
}
```

sorce code in S6 Class:

```
class S6 extends State {

    S6(StateMachine model) {
        super(model);
    }

    @Override
    void receipt() {
        if (model.s == model.LS[6]) {
            model.s = model.LS[0];
            model.getOP().PrintReceipt();
            model.getOP().ReturnCash();
        }
    }

    @Override
    void noReceipt() {
        if (model.s == model.LS[6]) {
            model.s = model.LS[0];
            model.getOP().ReturnCash();
        }
    }
}
```

Sorce code in StateMachine Class:

```
public class StateMachine {
    protected State s;
    protected State[] LS;
    private OutputProcessor op;

    public StateMachine() {
        LS = new State[8];
        //Pointer to the initial state
        LS[7] = new InitState(this);

        //Pointer to the S0 state
        LS[0] = new S0(this);

        //Pointer to the S1 state
        LS[1] = new S1(this);

        //Pointer to the S2 state
        LS[2] = new S2(this);

        //Pointer to the S3 state
        LS[3] = new S3(this);

        //Pointer to the S4 state
        LS[4] = new S4(this);
        //Pointer to the S5 state
        LS[5] = new S5(this);

        //Pointer to the S6 state
        LS[6] = new S6(this);

        //Pointer back to the initial state
        s = LS[7];
    }

    /*
        Getters and Setters
    */

    public OutputProcessor getOP() {
        return op;
    }

    public void setOP(OutputProcessor op) {
        this.op = op;
    }

    public void activate() {
        s.activate();
    }

    public void start() {
        s.start();
    }

    /*
        credit: t=1
        cash:   t=2
     */
    public void payType(int t) {
        s.payType(t);
    }

    public void approved() {
        s.approved();
    }

    public void reject() {
        s.reject();
    }

    public void cancel() {
        s.cancel();
    }

    /*
    Regular:    g=1
    Super:      g=2
    Premium:    g=3
 */
    public void selectGas(int g) {
        s.selectGas(g);
```

```java
    }

    public void startPump() {
        s.startPump();
    }

    public void pump() {
        s.pump();
    }

    public void stopPump() {
        s.stopPump();
    }

    public void receipt() {
        s.receipt();
    }

    public void noReceipt() {
        s.noReceipt();
    }
}
```