

# CSSE2003 Lecture 14 Supplement 2 (2009)

## Refactored Source

(from *Refactoring: improving the design of existing code*,  
Martin Fowler, Addison-Wesley, 2000)

### Movie.java:

```
public class Movie {
    public static final int REGULAR = 0;
    public static final int NEW_RELEASE = 1;
    public static final int CHILDRENS = 2;

    private String title;
    private Price price;

    public Movie (String title, int priceCode) {
        this.title = title;
        setPriceCode(priceCode);
    }
    public int getPriceCode() {
        return price.getPriceCode();
    }
    public void setPriceCode(int arg) {
        switch(arg){
            case REGULAR:
                price = new RegularPrice();
                break;
            case CHILDRENS:
                price = new ChildrensPrice();
                break;
            case NEW_RELEASE:
                price = new NewReleasePrice();
                break;
            default:
                throw new IllegalArgumentException("Incorrect Price Code");
        }
    }
    public String getTitle() {
        return title;
    }
    public double getCharge(int daysRented) {
        return price.getCharge(daysRented);
    }
    int getFrequentRenterPoints(int daysRented) {
        return price.getFrequentRenterPoints(daysRented);
    }
}
```

### Rental.java:

```
public class Rental {
    private Movie movie;
    private int daysRented;

    public Rental(Movie movie, int daysRented){
        this.movie = movie;
        this.daysRented = daysRented;
    }
    public int getDaysRented() {
        return daysRented;
    }
    public Movie getMovie() {
        return movie;
    }
    public double getCharge() {
        return movie.getCharge(daysRented);
    }
    public int getFrequentRenterPoints() {
        return movie.getFrequentRenterPoints(daysRented);
    }
}
```

Customer.java:

```
import java.util.*;

public class Customer {
    private String name;
    private List<Rental> rentals = new ArrayList<Rental>();

    public Customer(String name) {
        this.name = name;
    }
    public void addRental(Rental arg) {
        rentals.add(arg);
    }
    public String getName() {
        return name;
    }
    public String statement() {
        Iterator<Rental> rentalIterator = rentals.iterator();
        String result = "Rental Record for " + getName() + "\n";

        while (rentalIterator.hasNext()){

            Rental each = rentalIterator.next();

            //show figures for this rental
            result += "\t" + each.getMovie().getTitle() + "\t" +
                String.valueOf(each.getCharge()) + "\n";
        }
        // add footer lines
        result += "Amount owed is " + String.valueOf(getTotalCharge()) +
            "\n";
        result += "You earned " +
            String.valueOf(getTotalFrequentRenterPoints()) +
            " frequent renter points";
        return result;
    }
    private double getTotalCharge() {
        double result = 0.0;
        Iterator<Rental> rentalIterator = rentals.iterator();

        while (rentalIterator.hasNext()) {
            Rental each = rentalIterator.next();
            result += each.getCharge();
        }
        return result;
    }
    private int getTotalFrequentRenterPoints() {
        int result = 0;
        Iterator<Rental> rentalIterator = rentals.iterator();

        while (rentalIterator.hasNext()) {
            Rental each = rentalIterator.next();
            result += each.getFrequentRenterPoints();
        }return result;
    }
    public String htmlStatement() {
        Iterator<Rental> rentalIterator = rentals.iterator();
        String result = "<H1>Rentals for <EM>" + getName() +
            "</EM></H1><P>\n";

        while (rentalIterator.hasNext()){
            Rental each = rentalIterator.next();

            //show figures for this rental
            result += "\t" + each.getMovie().getTitle() + ": " +
                String.valueOf(each.getCharge()) + "<BR>\n";
        }
        // add footer lines
        result += "<P>You owe <EM>" + String.valueOf(getTotalCharge()) +
            "</EM><P>\n";
        result += "You earned <EM>" +
            String.valueOf(getTotalFrequentRenterPoints()) +
            "</EM> frequent renter points<P>";
        return result;
    }
}
```

## Price.java:

```
abstract class Price {
    abstract int getPriceCode();

    abstract double getCharge(int daysRented);

    int getFrequentRenterPoints(int daysRented){
        return 1;
    }
}
```

## RegularPrice.java:

```
class RegularPrice extends Price {

    int getPriceCode() {
        return Movie.REGULAR;
    }

    double getCharge(int daysRented) {
        double result = 2;
        if (daysRented > 2)
            result += (daysRented-2)*1.5;
        return result;
    }
}
```

## ChildrenPrice.java:

```
class ChildrensPrice extends Price {

    int getPriceCode() {
        return Movie.CHILDRENS;
    }

    double getCharge(int daysRented) {
        double result = 1.5;
        if (daysRented > 3)
            result += (daysRented-3)*1.5;
        return result;
    }
}
```

## NewReleasePrice.java:

```
class NewReleasePrice extends Price {

    int getPriceCode() {
        return Movie.NEW_RELEASE;
    }

    double getCharge(int daysRented) {
        return daysRented*3;
    }

    int getFrequentRenterPoints(int daysRented){
        return (daysRented > 1)? 2: 1;
    }
}
```