CIS 211

Design Patterns

Based on: http://csis.pace.edu/~bergin/patterns/strategydecorator.html
By Joseph Bergin, Pace University

Design Patterns

• Origin: "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides (Gang of Four) 1994.
• Common idioms for designing classes that work well together
• A Few Examples:
  – Abstract factory
  – Factory method
  – Singleton
  – Observer
  – Iterator
  – Strategy
  – Decorator
  – Composite
  – Adapter

Strategy Pattern

• Purpose: Separate out an algorithm that may vary from instance to instance or need to be selected at runtime.
• How It Works: Encapsulate algorithm as classes with a common interface to allow plug-and-play programming!
• Example: Suppose you want to read in a list of words and then print only...
  – Palindromes ("racecar", "bob", "madam")
  – or: Words that begin with 't' ("terrific", "tree", "Tarzan")
  – or: Words that have exactly four letters ("bugs", "love", "code")
How can we minimize redundant code?

Example by: Joseph Bergin, Pace University, jbergin@pace.edu

Case 1: Palindromes

public void printWhenPalindromes(String filename) throws IOException {
  Scanner in = new Scanner(new FileReader(filename));
  // Read in lines one at a time
  while (in.hasNextLine()) {
    String currLine = in.nextLine();
    StringTokenizer words = new StringTokenizer(currLine);
    while (words.hasMoreTokens()) {
      String word = words.nextToken();
      // Print line only if it contains a matching word
      if (isPalindrome(word)) {
        System.out.println(currLine);
        break;
      }
    }
  }
}

Example by: Joseph Bergin, Pace University, jbergin@pace.edu
Case 2: Begins with 't'

```java
public void printWhenBeginsWithT(String filename) throws IOException {
    Scanner infile = new Scanner(new FileReader(filename));
    // Read in lines one at a time
    while (scanner.hasNextLine()) {
        String currLine = scanner.nextLine();
        StringTokenizer words = new StringTokenizer(currLine);
        while (words.hasMoreTokens()) {
            String word = words.nextToken();
            // Print line only if it contains a matching word
            if (word.charAt(0) == 't' || words.charAt(0) == 'T') {
                System.out.println(currLine);
                break;
            }
        }
    }
}
```

Case 3: Four letter words

```java
public void printFourLetterWords(String filename) throws IOException {
    Scanner infile = new Scanner(new FileReader(filename));
    // Read in lines one at a time
    while (scanner.hasNextLine()) {
        String currLine = scanner.nextLine();
        StringTokenizer words = new StringTokenizer(currLine);
        while (words.hasMoreTokens()) {
            String word = words.nextToken();
            // Print line only if it contains a matching word
            if (word.length() == 4) {
                System.out.println(currLine);
                break;
            }
        }
    }
}
```

Attempt 1

```java
public void printWhen(String filename, int choice) throws IOException {
    Scanner infile = new Scanner(new FileReader(filename));
    // Read in lines one at a time
    while (scanner.hasNextLine()) {
        String currLine = scanner.nextLine();
        StringTokenizer words = new StringTokenizer(currLine);
        while (words.hasMoreTokens()) {
            String word = words.nextToken();
            // Print line only if it contains a matching word
            if (choice == 1) {
                if (isPalindrome(word)) {
                    System.out.println(currLine);
                    break;
                }
            } else if (choice == 2) {
                if (word.charAt(0) == 't' || words.charAt(0) == 'T') {
                    System.out.println(currLine);
                    break;
                }
            } else { // (choice == 3)
                if (words.length() == 4) {
                    System.out.println(currLine);
                    break;
                }
            }
        }
    }
}
```

Strategy Pattern

```java
public void printWhen(String filename, CheckStrategy strategy) throws IOException {
    Scanner infile = new Scanner(new FileReader(filename));
    // Read in lines one at a time
    while (scanner.hasNextLine()) {
        String currLine = scanner.nextLine();
        StringTokenizer words = new StringTokenizer(currLine);
        while (words.hasMoreTokens()) {
            String word = words.nextToken();
            // Print line only if it contains a matching word
            if (strategy.check(word)) {
                System.out.println(currLine);
                break;
            }
        }
    }
}
```
Strategy Implementation (1)

public interface CheckStrategy
{
    public boolean check(String s);
}

public class StartsWithT implements CheckStrategy
{
    public boolean check(String s)
    {
        return s.charAt(0) == 't' || s.charAt(0) == 'T';
    }
}

public class FourLetters implements CheckStrategy
{
    public boolean check(String s)
    {
        return s.length == 4;
    }
}

printWhen("hamlet.txt", new StartsWithT());
printWhen("hamlet.txt", new FourLetters());

Strategy Implementation (2)

public interface CheckStrategy
{
    public boolean check(String s);
}

public class NumLetters implements CheckStrategy
{
    private int numLetters;
    public NumLetters(int numLetters)
    {
        this.numLetters = numLetters;
    }
    public boolean check(String s)
    {
        return s.length == numLetters;
    }
}

printWhen("hamlet.txt", new NumLetters(5));

Decorator Pattern

- **Purpose:** Add functionality to an existing class.
- **How It Works:** Contains the original object, but adds some new functionality.

Example by: Joseph Bergin, Pace University, jbergin@pace.edu
Decorator Pattern

• **Purpose:** Add functionality to an existing class.
• **How It Works:** Contains the original object, but adds some new functionality.
• **Example:** Suppose you want to count the words while checking to see if they satisfy the criterion. How can we accomplish this?

• Other Examples:
  – Adding a peek() method to a Queue class
  – Adding scroll bars to a Window class

Example by: Joseph Bergin, Pace University, jbergin@pace.edu

Decorator Example

class CounterDecorator implements CheckStrategy {
    private int count = 0;
    private CheckStrategy checker;
    public CounterDecorator(CheckStrategy check) {
        checker = check;
    }
    public boolean check(String s) {
        boolean result = checker.check(s);
        if (result) count++;
        return result;
    }
    public int count() { return count; }
}

CounterDecorator counter = new CounterDecorator(new NumLetters(5));
printWhen("hamlet.txt", counter);
System.out.println("Hamlet has "+ counter.count + " lines with 5-letter words.");

Composite Pattern

Suppose we want 5-letter-words starting with t:

class AndStrategyComposite implements CheckStrategy {
    private List<CheckStrategy> tests = new LinkedList();
    public void addStrategy(CheckStrategy s) {
        tests.add(s);
    }
    public boolean check(String s) {
        for (CheckStrategy strategy: tests) {
            if (!strategy.check(s)) return false;
        }
        return true;
    }
}

CheckStrategy tAndFive = new AndStrategyComposite();
tAndFive.add(new BeginsWithT());
tAndFive.add(new NumLetters(5));
printWhen("hamlet.txt", tAndFive);

Bonus: Adapter Pattern

• **Question:** Suppose we don’t have a PriorityQueue<E> class and want to create one using a TreeMap<E>. How?

• **Answer:** Use a TreeMap<E> internally to implement the PriorityQueue<E> methods (add, remove).

• **Alternate Answer:** Inherit from TreeMap<E> and implement Queue interface.