Mathematisch–Naturwissenschaftliche Fakultät
Fachbereich Informatik (WSI)
Praktische Informatik
Prof. Dr. Klaus Ostermann

Jonathan Brachthäuser

November 21, 2014

Homework Assignment 5

Software Design & Programming Techniques (WS2014)

The deadline for this assignment is Nov 26, 23:59. Hand in the assignments via email to jonathan.brachthaeuser@uni-tuebingen.de. Please include all names of the group members in your email.

1 Decorator Pattern

Answer the following questions regarding the Interface Segregation Principle (ISP):

1. What is the Decorator Pattern? Shortly describe “Intent”, “Solution” and “Consequences” for this pattern.
2. The behavior of class instances can also be influenced by subclassing and overwriting methods. Create a table that compares Inheritance with the Decorator Pattern. What are the individual benefits and problem. When should you use which solution?
3. Design the following specification in a statically typed, OO language of your choice. (Do not spend time on the actual implementation of the components. Classes, interfaces and method signatures are enough. Sometimes it is helpful to sketch the body of a method.)

   A text-view component should be able to display the contained text on the screen. Some content might be so large that it should be possible to add scroll-bars to individual text-view components. Text-views extended with scroll-bars should be able to scroll to a specific position in the text. Border around text-views are “nice” to visually highlight some of them.

4. Let us assume we can use your visual component library like the following:

```
tv = new TextView
tv.display()
AddScrollBars(tv).scrollTo(20).display()
```

   (a) Can you add a border around a scrollable component and still scroll it? In code:

   ```
   AddBorder(AddScrollBars(tv)).scrollTo(20)
   ```

   (b) Assume you collect all text-views in a set called *tvs* that uses object identity to compare the text-views. What will the following code return? Explain your answer.

   ```
   tvs.add(tv)
   tvWithScrollbar = AddScrollBars(tv)
   println(tvs.contains(tvWithScrollbar))
   ```

   (c) Assume that the original text-view implementation uses the method *calcInnerWidth* to implement the displaying functionality. Adding a scroll-bar reduces the area to show content. Why does overriding *calcInnerWidth* in the scroll-bar decorator not lead to the correct results?

   (d) Do you have ideas on how to fix the problems from the previous questions (a-c)?