APIT - patterns - lab 2

Simon Rogers

4th March 2015

Introduction and aims

So far, we've used individual patterns. In this lab you will combine both the `composite` and `decorator` patterns.

Tasks

1. In the lectures I went through a shop example. You should start by implementing a slight variation on this using the `composite` pattern. The shop has items (leaves) and also sells groups of items (composites). Items have `name` and `price` attributes, composites just have a `name` attribute (their price is computed through looping over their children). Ignore the discount stuff in the lecture example for now. Composites should be able to include composites. All of the components in the system should implement a `compPrice()` method and `toString()` method.
2. Now we will implement the discounts but using the `decorator` pattern. Make a `StudentDiscountDecorator` that discounts by 10% and a `StaffDiscountDecorator` that discounts by 50%. Some hints:
   (a) The highest level interface for the `decorator` you already have from the `composite`.
   (b) You will need an additional `abstract` class (e.g. `ShopComponentDecorator`) and then concrete `decorator` classes for the two `decorators`
   (c) It should be possible to `decorate` both leaves and composites!