The Visitor Pattern

Relevant Links

- OODesign[1]
- GoF Book[2], page 163

Notes

We discuss the Composite pattern, which enables complicated structures of items and groupings of those items.

Classification

The Composite is a Structural Pattern.

Intent

The Composite Pattern's intent is to compose objects into tree structures to represent part-whole hierarchies. It lets clients treat individual objects and compositions of objects uniformly.

Motivation

Imagine a graphics application. We would have primitives like Line, Rectangle, Text etc, but we also want to compose them together to form more complicated shapes, like a Picture.

These composed elements must contain some extra methods like a way to access their "children". But at the same time clients need to be able to access "graphical elements" without needing to know if they are primitives or composites.

This is done by having both primitives and composites implement the same *interface* or *abstract class*, say a "Graphic" interface. That interface will contain the methods that are shared by both primitive and composite elements. On top of that, the composite elements need to have operations for managing their children.

[1] http://www.oodesign.com/composite-pattern.html
[2] http://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented/dp/0201633612/

Participants / Implementation

Component This is the abstract class or interface that primitive and composite elements share. It may also implement default behavior when appropriate.

Leaf These are the primitive objects. They have no children.

Composite Composites define the behavior for the components that have children and implement the related operations.

Client Manipulates the objects in the collection through the Component interface. The goal of the pattern is to allow clients to work with the objects in the collection without needing to know if they are leaves or composites.

The Composite Pattern can work well with Iterator and Visitor, as we will see in the examples.

An important consideration is how rich to make the Component interface:

- Should it have default implementations for most leaf methods?
- Should it have default implementations for child management methods?

This is even more of a concern in a statically typed setting.

Javascript implementation See composite.js[3]

[3] ../../testPages/composite.js