Visitor

Problem

You have an object structure (tree, list, graph, etc.), and several algorithms that traverse the object structure and process each node in some way. The object structure contains many different kinds of nodes.

Object structure + Traversal code + Algorithm-specific node processing

Example: RGB node example (src-original)

Disadvantages:

1) Code for algorithms is distributed across node classes. Would prefer to have all the code for each algorithm on one class.
2) Code duplication. Traversal code is duplicated in each algorithm.
3) When a new algorithm needs to be added, all node classes must be modified.

Solution: Visitor (src-with-visitor)

This solution still duplicates the traversal code in each node class. In this case, we could centralize the traversal code on the base class to avoid this duplication. (This is only possible because all node types implement the same child management interface.)

Solution: Visitor (src-with-visitor-and-shared-traversal)

General Visitor class diagram (slides)

Visitor homework