Visitor

3-30-2004

Opening Discussion

- Make defensive copies of mutable objects to defend your copy. This isn't needed with immutable objects since they can't be changed.
- Don't overload with ambiguous types. If you have two versions that take subtype and supertype, the version is picked statically. This isn't likes overriding which works dynamically.
- Return an object not a reference when the method creates an object.

More Discussion

- There are two main cases where ambiguity can be created. One is when you overload with types that can be reached equally well from implicit conversions. The other is with multiple inheritance from supertypes that have some name in common.
- Do you have any questions about the reading for today?

Visitor

- This pattern provides a way to perform an operation on all of the elements of some structure. This allows you to change the operation without having to alter the code for the structure itself.
- To do this we define a Visitor interface with a method that is called when an object is “visited” that method also accepts the object being visited as an argument.
- The ConcreteVisitor can collect information, or might be like a simple functor.

Example

- GoF uses an example of a compiler where the code is compiled to a meta-format and then we pass through that format several times doing different things.
- Without visitor, we need a different method for each type of pass we want to do. That adds a new method to every type in our meta-format. With visitor we just create a new ConcreteVisitor.
- More familiar to you would be having a binary tree and wanting to do different things to the elements.

Benefits and Drawbacks

- The primary benefit is that it is very easy to add new operations. It is also easy to change that functionality for a single visitor because it sits in a single visitor class.
- This also keeps things related to different operations separated into different visitors.
- When you want to add a new type to the structure, that can be hard. This is because different visitors might handle each subtype differently.

More Benefits and Drawbacks

- Visitors can work across class hierarchies. An iterator can't do this. The visitor can also be more efficient than an iterator in some cases.
- The visitor can accumulate state where data would have to be passed through if the methods were part of the objects.
- Visitors often require you to break encapsulation in some way.

Progress Presentations

- Bobby and Pete are going to present today. That leave 5 people who haven't presented and there are some people who might want to go again. We need two people going every day from here out.