.field private elements [Ljava/lang/Object;
.field current I

.method public getNext()Ljava/lang/Object;
;; Return elements[current], and increment current
.end method

.method public anyMore()Ljava/lang/Object;
;; Return false if current < the length of elements
.end method

The implementor of Business can change the implementation of inventory without altering any of the classes that use it, since the implementation of Enumerator was kept separate from the interface definition.

Invoking methods through interfaces is slightly different from invoking methods using invokevirtual or invokespecial. A third instruction, invokeinterface, is required. To call anyMore, use code like this:

;; Assume there's a Business in local variable 0
aload_0 ; Get the Business
invokevirtual Business/inventory ()LEnumerator; ; Get its inventory
invokeinterface Enumerator/anyMore ()Z 1 ; Call anyMore

The last argument to the invokeinterface instruction is the number of stack words used as parameters to the method call, including the receiver itself. In this example, the only parameter is the Enumerator object itself, so the value is 1. For the interface:

.interface Searchable
.method find(Ljava/lang/String;)Ljava/lang/Object;
.end method

A call to find looks like this:

;; Assume there's a Searchable object in register 1
aload_1 ; Get the object
ldc "potato chips" ; We want an element named "potato chips"
invokeinterface Searchable/find
(Ljava/lang/String;)Ljava/lang/Object; 2

2 This number should not be necessary, since it can be derived from the method descriptor. However, it is included in the Oolong language because it is part of the underlying JVM bytecodes.