Activity 16-2: The Iterator Pattern

Why?

The Iterator pattern is one of the most pervasive but also one of the simplest of the mid-level design patterns. It serves as a good example of such patterns.

Learning Objectives

- Understand the Iterator pattern and how to use it to traverse a collection
- Understand how to realize iteration mechanisms

Success Criteria

- Be able to use iterators to traverse a collection
- Be able to explain the static and dynamic structure of the Iterator pattern
- Be able to design and code iteration mechanisms with and without using the Iterator pattern

Resources

ISED section 16.2

Vocabulary

Iterator, internal and external iteration control, built-in iteration mechanisms

Plan

1. Review *ISED* section 16.2 individually.
2. Answer the Key Questions individually, and then evaluate the answers as a team.
3. Do the Exercises as a team, and check your answers with the instructor.
4. Do the Problems and Assessment as a team.
5. Turn in the Problems and Assessment as a team deliverable.

Key Questions

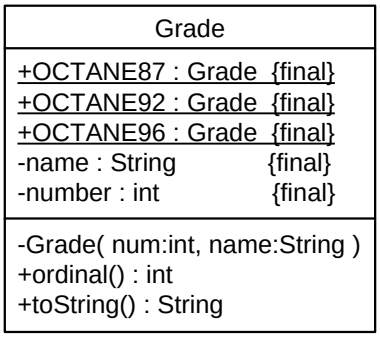
1. What is an iterator?
2. What four operations must every iterator implement?
3. Why are iterators used?

Exercise

1. Do clients or collection create iterators in the Iterator pattern?
2. Why do Java iterators only have two operations (besides the optional `remove()` operation)?

Problems (Deliverable)

A gas station simulation program must keep track of gasoline grades. A Grade enumeration is used for this purpose and implemented as a Java type-safe enumeration. In this implementation, the Grade class (pictured below) is instantiated to each of the enumeration values as static final attributes, and the Grade() constructor is private so that no new enumeration values can be created. Variables holding enumeration values are then constrained to hold only the declared values of the Grade enumeration.



1. The Grade class, as an enumeration, can be thought of as a collection holding values (the three grades). As such, it may be traversed. Design a Grade enumeration external iterator based on the Iterator pattern.
   (a) Does your design use the Java Iterator or Collection interfaces? Why or why not?
   (b) Make a class diagram depicting your design.
   (c) How must your design differ from Figure 16.2.1 in ISED section 16.2, and why?
   (d) How can the iterator access the collection (the enumeration) and its elements?
2. Design a **built-in** external iteration mechanism for the Grade class.
   (a) Does your design use the Java Iterator or Collection interfaces? Why or why not?
   (b) Make a class diagram depicting your design.
3. Compare and contrast your designs. What advantages and disadvantages does each one have? Which alternative is best?

Assessment (Deliverable)

1. Did this activity help you achieve the learning objectives?
2. How could the instructor improve this activity?