Iterator Design Pattern

A Behavioral Pattern
By Michael Walt
Intent

- Provide a way to access aggregate objects sequentially without exposing their underlying structure.
- Allow for different traversal methods.
- Allow for concurrent traversals of the same object.
- Use to avoid breaking encapsulation by requiring data access through iterator only.
- Provide a means for polymorphic traversal.
Aggregate Objects

- ArrayList
- LinkedList
- HashTable
- Stack
- TreeSet
- EnumSet, LinkedHashSet, PriorityQueue, ..., Composite or Custom Aggregates
Common Use Case

TV Remote Control:
- Uses current channel to determine next (up arrow) and previous (down arrow) channels:
  
  ```java
  public interface Iterator {
  public Channel nextChannel(int currentChannel);
  public Channel prevChannel(int currentChannel);
  }
  ```

- Specifically, the remote provides functionality for “surfing” channels by implementing the Iterator interface:
  ```java
  public ChannelSurfer implements Iterator {
  /*** nextChannel – method which takes the current channel number and returns the next channel.*/
  public Channel nextChannel (int currentChannel)
  {Channel channel = new Channel(currentChannel+1); return channel; } 
  /*** prevChannel – method which takes the current channel number and returns the previous channel.*/
  public Channel prevChannel (int currentChannel)
  {Channel channel = new Channel(currentChannel-1);return channel; }
  ```

- Likewise, the remote can be programmed to implement the iterator to return the programs straight away rather than the channel – like the channel menu in comcast, directTV, and RCN
TV Remote UML

Aggregate

ChannelFrequencies
+methodOfTraversal()

TunedChannel
+methodOfTraversal()

ChannelSelector

Iterator

ChannelIterator
+next()
+previous()

Concrete Aggregate

Concrete Iterator
Participants

- Iterator – defines the interface to access and traverse elements
- ConcreteIterators – implements the iterator and keeps track of the current position in the traversal
- Aggregate – defines the interface for creating the iterator object (using factory pattern)
- ConcreteAggregate – implements the iterator creation interface to return the reference to the Concretelterator.
UML Diagram 1

```
Aggregate
  + iterator()

<< create >>

Iterator
  + next()
  + hasNext()

<< create >>

ConcreteAggregate
  + iterator()

ConcreteIterator
  + next()
  + hasNext()
```
Sample Java Code

- See Eclipse IteratorDemo
References

- http://sourcemaking.com/design_patterns/iterator
- http://sourcemaking.com/design_patterns/iterator/java/1