

Iterator Pattern

집합체의 요소들을 동일한 방법으로, 순서대로 처리.

[Architecture](#), [Modeling](#), [Design Pattern](#), [Behavioral](#)

반복자(Iterator) API

```
public interface Iterator<T> {  
    boolean hasNext();  
    T next();  
}
```

```
public class ItemSequenceIterator<T> implements Iterator<T> {  
    private ItemSequence<T> concreteAggregate;  
    private int index = 0;  
  
    public ItemSequenceIterator(ItemSequence<T> concreteAggregate) {  
        this.concreteAggregate = concreteAggregate;  
        this.index = 0;  
    }  
  
    @Override  
    public boolean hasNext() {  
        return index < concreteAggregate.length();  
    }  
  
    @Override  
    public T next() {  
        T item = concreteAggregate.get(index);  
        index++;  
        return item;  
    }  
}
```

반복할 집합체(Aggregate)

```
public interface Aggregate<T> {  
    Iterator<T> iterator();  
}
```

```
/* 간단한 ArrayList를 구현한 형태 */
public class ItemSequence<T> implements Aggregate<T> {
    private T[] items;
    private int last = 0;

    public ItemSequence(int capacity) {
        this.items = (T[]) new Object[capacity];
    }

    public T get(int index) {
        return items[index];
    }

    public void add(T item) {
        this.items[last] = item;
        last++;
    }

    public int length() {
        return last;
    }

    @Override
    public Iterator<T> iterator() {
        return new ItemSequenceIterator<T>(this);
    }
}
```

Client

```
public class Main {
    public static void main(String[] args) {
        ItemSequence<Item> itemContainer = new ItemSequence<>(2);
        itemContainer.add(new Item("Item1"));
        itemContainer.add(new Item("Item2"));
        Iterator<Item> it = itemContainer.iterator();
        while(it.hasNext()) {
            Item item = it.next();
            System.out.println(item.getStr());
        }
    }
}

/* 출력 */
//Item1
//Item2
```

From:

<http://ledyx.xyz/> - **Ledyx WIKI**

Permanent link:

http://ledyx.xyz/doku.php?id=design_pattern:iterator_pattern

Last update: **2021/02/07 03:26**

