Iterator 패턴

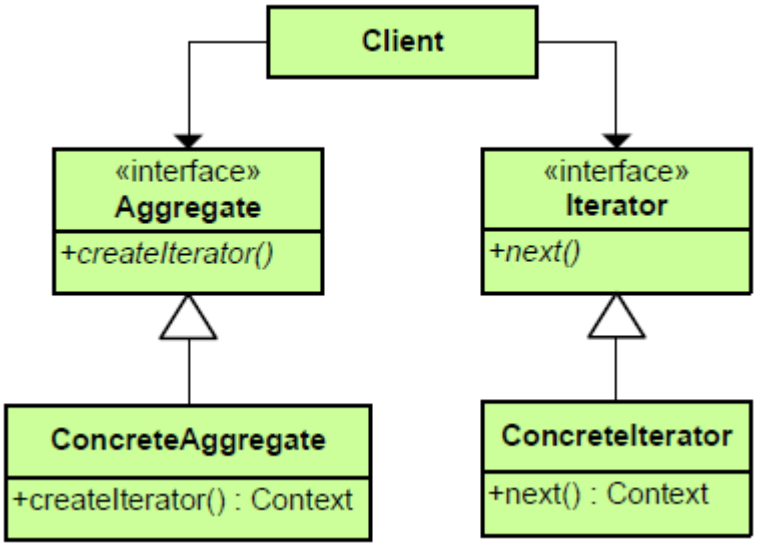
타입

Behavioral Pattern

문제

해결

클래스 다이어그램



예제

iterator.cpp

```
#include <iostream>
using namespace std;

class Stack
{
    int items[10];
    int sp;
  public:
    friend class StackIter;
    Stack()
    {
```

```
        sp =  - 1;
    }
    void push(int in)
    {
        items[++sp] = in;
    }
    int pop()
    {
        return items[sp--];
    }
    bool isEmpty()
    {
        return (sp ==  - 1);
    }
    StackIter *createIterator()const; // 2. Add a createIterator()
member
};

class StackIter
{
    // 1. Design an "iterator" class
    const Stack *stk;
    int index;
  public:
    StackIter(const Stack *s)
    {
        stk = s;
    }
    void first()
    {
        index = 0;
    }
    void next()
    {
        index++;
    }
    bool isDone()
    {
        return index == stk->sp + 1;
    }
    int currentItem()
    {
        return stk->items[index];
    }
};

StackIter *Stack::createIterator()const
{
  return new StackIter(this);
}
```

```
bool operator == (const Stack &l, const Stack &r)
{
  // 3. Clients ask the container object to create an iterator object
  StackIter *itl = l.createIterator();
  StackIter *itr = r.createIterator();
  // 4. Clients use the first(), isDone(), next(), and currentItem()
protocol
  for (itl->first(), itr->first(); !itl->isDone(); itl->next(),
itr->next())
    if (itl->currentItem() != itr->currentItem())
      break;
  bool ans = itl->isDone() && itr->isDone();
  delete itl;
  delete itr;
  return ans;
}

int main()
{
  Stack s1;
  for (int i = 1; i < 5; i++)
    s1.push(i);
  Stack s2(s1), s3(s1), s4(s1), s5(s1);
  s3.pop();
  s5.pop();
  s4.push(2);
  s5.push(9);
  cout << "1 == 2 is " << (s1 == s2) << endl;
  cout << "1 == 3 is " << (s1 == s3) << endl;
  cout << "1 == 4 is " << (s1 == s4) << endl;
  cout << "1 == 5 is " << (s1 == s5) << endl;
}
```

참고

http://en.wikibooks.org/wiki/C%2B%2B_Programming/Code/Design_Patterns#Iterator