Iterator

Propósito

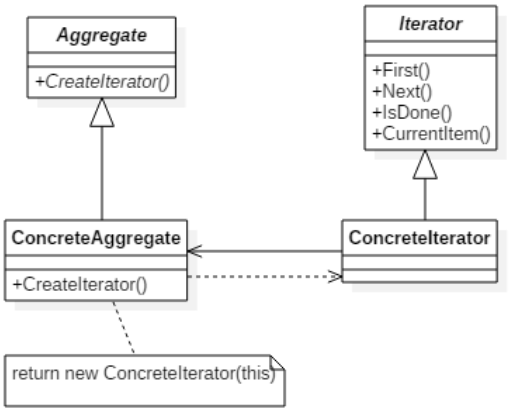
Proporciona una forma de acceder secuencialmente a los elementos de una colección (aggregate) sin exponer su implementación.

Aplicabilidad

El patrón Iterator se utiliza:

- Para acceder al contenido de una colección sin exponer su representación interna
- Para soportar múltiples formas de recorrer una colección
- Para proporcionar una interface uniforme para recorrer diferentes estructuras (para soportar una iteración polifórmica)

Estructura



Participantes

- Iterator: define la interface para acceder y recorrer los elementos
- ConcreteIterator:
    - implementa la interface Iterator.
    - Guarda la posición actual en la colección
- Aggregate: define la interface para crear un objeto Iterator
- ConcreteAggregate: implementa la interface de Aggregate para devolver el objeto ConcreteIterator adecuado

Colaboraciones

- Un ConcreteIterator mantiene una referencia al objeto actual de la colección y es capaz de calcular el siguiente.

Consecuencias

El patrón Iterator tiene tres consecuencias:

1. Soporta variantes en cuanto a la forma de recorrer la colección. Las colecciones complejas se pueden recorrer de muchas maneras. El Iterator permite modificar de forma sencilla el algoritmo utilizado para recorrer la colección, basta con definir otro ConcreterIterator.
2. Los Iterators simplifican la interface de Aggregate. Al tener el Iterator, eliminamos la necesidad de tener esas operaciones en Aggregate, con lo cual se simplifica su interface.
3. Se puede tener más de recorrido de la colección a la vez. Como cada Iterator almacena su propio “elemento actual”, podemos manejar más de un recorrido diferente a la vez.