Patrón de diseño:

Iterator

ITERATOR

Integrantes: Alvarez Nestor
Daniotti Danilo
Etcheverry Pablo

Introducción

Un patrón de diseño es una descripción de clases y objetos que interactúan entre si, que son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software.

Proporcionan una manera de resolver un problema, evitan perder el tiempo en soluciones a problemas ya resueltos o conocidos, crean código reusable.

Ayudan a resolver problemas de una manera estándar y profesional

Los patrones de diseño se clasifican en
- Patrones de creación
- Patrones de estructura
- Patrones de comportamiento

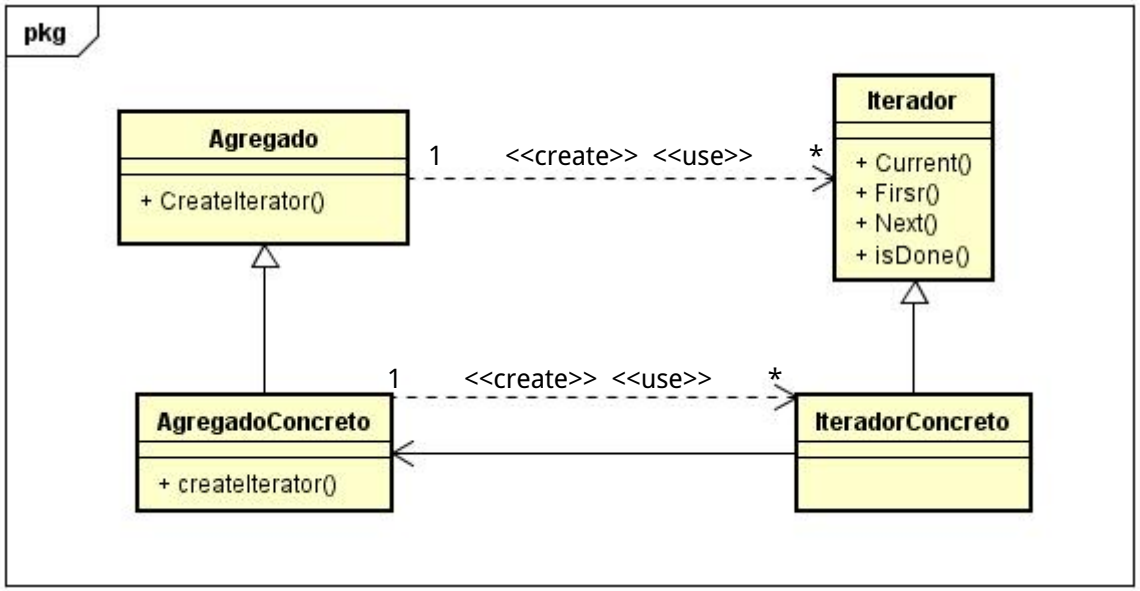
ITERATOR

Iterator

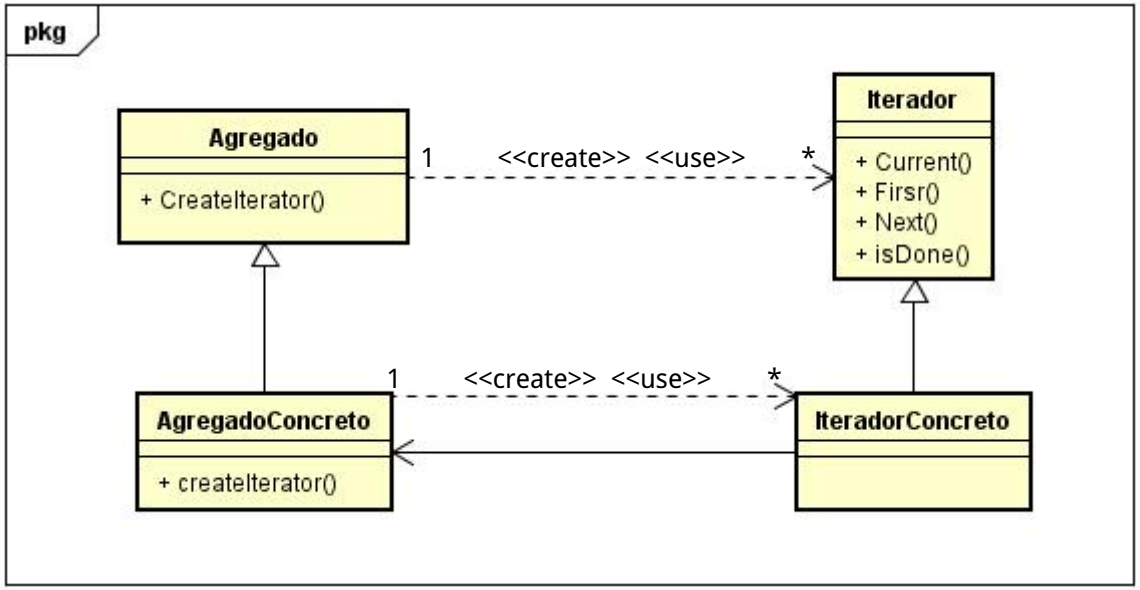
El iterator es un patrón de comportamiento basado en objetos que brinda un modo de acceder secuencialmente a los elementos de un objeto compuesto sin exponer su representación interna (implementación)

ITERATOR

Estructura



ITERATOR

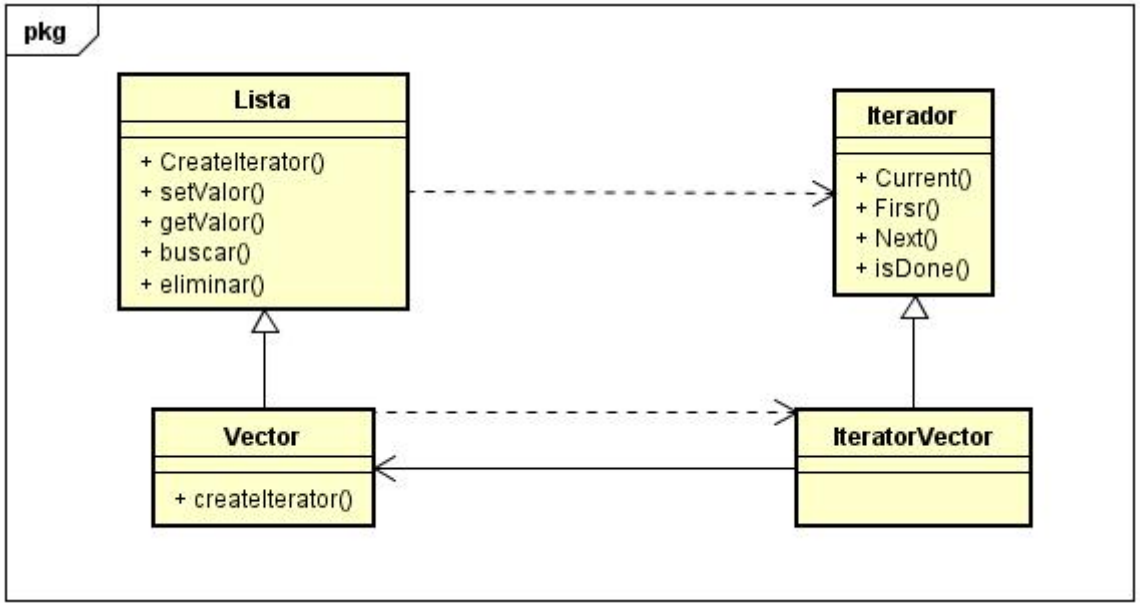


Participantes

- **Iterador**
  Define una interfaz para recorrer los elementos y acceder a ellos
- **IteradorConcreto**
  Implementa la interfaz iterador
  Mantiene la posición actual en el recorrido del agregado
- **Agregado**
  Define una interfaz para crear un objeto iterador
- **AgregadoConcreto**
  Implementa la interfaz de creación de Iterador para devolver una instancia del iterador concreto apropiado

Ejemplo

- En el siguiente ejemplo se implementa una lista mediante un Vector del cual se muestran sus componentes haciendo uso de la clase Iterator.



ITERATOR

Código

ITERATOR

```java
public interface Lista<T> {
    public Iterator createIterator();
    public void setValor(T e, int i);
    public T getValor(int i);
    public int buscar();
    public void eliminar(int i);
}
```

```java
public interface Iterator<T> {
    public T current();
    public T first();
    public void next();
    public boolean isDone();
}
```

```java
class Vector implements Lista<Integer> {
    private int[] vector;

    public Vector(int n) {
        vector = new int[n];
    }

    public IteratorVector createIterator() {
        return new IteratorVector(this);
    }
    public void setValor(Integer e, int i) {
        vector[i] = e;
    }
    public Integer getValor(int i) {
        return vector[i];
    }
```

```java
class IteratorVector implements Iterator {
    int[] vector;
    int pos;

    public IteratorVector(Vector v) {
        vector = v.getArray();
        pos = 0;
    }

    public Integer current() {
        return vector[pos];
    }
    public Integer first() {
        return vector[0];

    }
    public void next() {
        pos++;
    }
    public boolean isDone() {
        return pos == vector.length;
    }
}
```

```java
class Main {

    public static void main(String[] args) {
        Vector test = new  Vector(3);
        // Creacion del iterator
        IteratorVector iteratorTest = test.createIterator();
        // Recorrido del iterator
        while (!iteratorTest.isDone()) {
            System.out.println(iteratorTest.current());
            iteratorTest.next();
        }

    }

}
```