Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Práctica 8

Enunciado de la tarea.

Enrique Suárez Hernández - alu0101325781@ull.edu.es

C/ Padre Herrera s/n
38207 La Laguna
Santa Cruz de Tenerife. España

T: 900 43 25 26

ull.es

Introducción

En esta práctica se nos marca la tarea de eliminar todos los usos de un desarrollo propio de arrays que se usa en diferentes implementaciones y a partir de ahora usar siempre la clase Lista de java. Para conseguir este objetivo sin realizar modificaciones sustanciales en el código existente se decidió usar el patrón adaptador.

También se pensó en utilizar el patrón estrategia ya que permite la existencia de varias implementaciones en caso de que la empresa decida volver en algún momento al uso de arrays, pero ya que se nos especifica explícitamente que siempre van a usar la clase Lista se decidió trabajar con el adaptador.

Este patrón de diseño proporciona una solución intermedia al encapsular la lógica de adaptación, permitiendo una transición gradual. Pudiendo reemplazar progresivamente instancias de la antigua implementación con instancias del adaptador en proyectos existentes. Además garantiza la independencia entre las implementaciones, ya que encapsula la lógica de adaptación.

Diagrama UML

Siguiendo el patrón mencionado anteriormente se ha creado la clase AdapterList, (Figura 1) siendo su objetivo principal actuar como un puente entre la interfaz existente (MiArray) y List.

De esta forma, puede ser utilizada en lugar de las implementaciones en la clase MiArrayClase proporcionada, sin que el cliente note la diferencia. Cada método tiene una implementación que utiliza internamente la clase List para realizar las operaciones correspondientes. Traduciendo las llamadas de los métodos de MiArrayClase a las operaciones equivalentes.

También se consideró oportuno la creación de IVentana, una interfaz que crea una ventana para probar los métodos de AdapterList.

Donde se puede añadir números a la lista, eliminarlos, vaciarla, comprobar si está vacía, devolver el primer valor, el último y su posición en la lista.

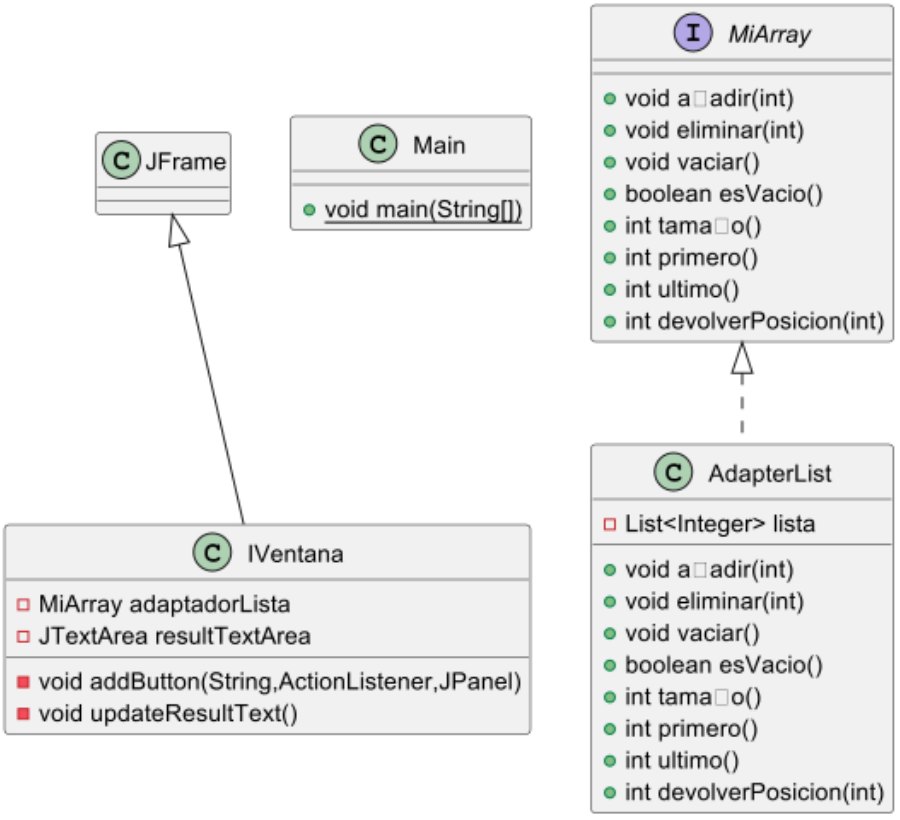

Figura 1 : Diagrama UML

Tabla de tiempo

	Tiempo esperado	Tiempo real
Programa	1h	1h

Para la realización de esta práctica donde más dificultades encontré fue a la hora de plantear qué patrón de diseño se adaptaría mejor a las necesidades del programa. Después de informarme encontré que el mejor sería el patrón adaptador. (20 minutos)

Posteriormente a esto, la implementación de código fue sencilla puesto que era la creación de una clase, que utilizara List, para adaptar los métodos de MiArrayClase. (20 minutos)

Una vez conseguida la clase AdapterList se comenzó con la creación de este informe para su posterior entrega. (20 minutos).

Bibliografía

Refactoring Guru