Yalbani Aranda

INYECCION POR DEPENDENCIAS

A un mesero se le va a asignar una orden de comida que va a entregar en un restaurante.

Es un funcionamiento de *bajo acoplamiento*, ya que un cocinero va a ser el responsable (inyector) de que se le asigne el tipo de platillo que va a preparar. El mesero solo va a conocer la orden (dependencias), definida a través de una *interface*. Entonces, las ordenes o dependencias pueden intercambiarse con una implementación distinta sin que los objetos que dependen de ellas conozcan las diferencias entre unas y otras.

El proyecto implementa pilares de la POO, como polimorfismo, abstracción, herencia y encapsulamiento. Podemos observar la herencia en la sobre escritura, la abstracción con la implementación de interfaces, y el encapsulamiento en el uso de modificadores de acceso y *getters* y *setters*. El polimorfismo lo podemos observar en como el mensaje (orden) puede tener distinto comportamiento, dependiendo de lo que el inyector(cocinero) le entregue al mesero.

INYECCION POR VARIABLE

La interface le va a informar que tipo de platillo va a entregar de manera abstracta, no como se preparó, de manera concreta. En cada método se está sobrescribiendo el método original. En *main,* o en la barra del restaurante, se crea el objeto mesero y se le asigna un nombre para diferenciar a los meseros.

El mesero solo sabe que tiene asignada una orden para entregar, pero no tiene la responsabilidad de prepararla. Así se le inyecta a través de una **variable de referencia** la orden y su método de entregarla. La clase Inyector es el agente externo que va a tener la responsabilidad de inyectarle la orden al mesero a través de la variable de referencia, por medio de un setter, que le asigna esa orden solo a ese mesero.

INYECCION POR CONSTRUCTOR

El mesero en su constructor conoce la abstracción de la orden, pero no lo que va a contener la orden. De igual manera, quien toma esa responsabilidad es el inyector a través del constructor.

El objeto Mesero se inyecta directamente en *main* y la orden se le inyecta en el constructor al Mesero. El inyector regresa un Mesero al que asignarle la orden; entonces en *main* ya no solo se crea el objeto Mesero, sino que el mesero ya fue identificado y definido por el inyector y recibe directamente la orden.

La diferencia entre ambas es que, en la inyección por constructor, mediante el constructor ya se define un mesero y la orden que se le va a asignar desde el Inyector. La orden ya está lista, se busca un mesero en *main* para entregarla. En la inyección por variable, se le asigna a un mesero ya definido, una variable de referencia que contiene la orden, y con un setter se le informa al mesero que orden va a ser al inyectársela en *main*, se llama al mesero y se le da la orden que ya está lista.

Me parece que la inyección por Constructor es más práctica, ya que tiene código más limpio, es simple y modular. Permite entender más fácilmente el código, y encontrar más fácilmente las dependencias.