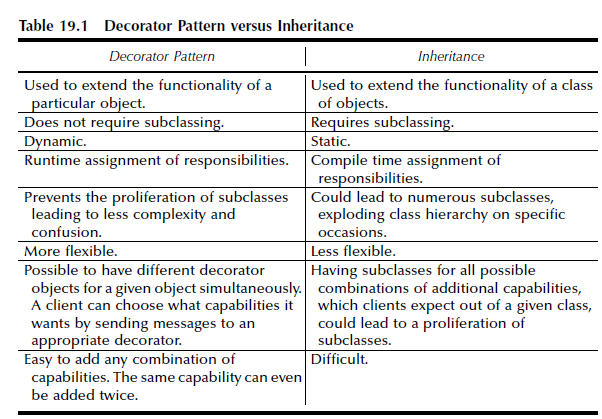
# El patrón de diseño decorador

Este patrón tiene como finalidad extender funcionalidad de un objeto sin realizar modificaciones a la clase o recurrir a herencia. Esto se logra a través de un objeto de envoltura “wraper” u objeto Decorador al rededor del objeto actual.

Características del Decorador:

* El objeto decorador está diseñado para tener la misma interfaz que el objeto base. De esta manera el cliente interactúa con el objeto decorador de la misma manera que si lo hiciera con el objeto base.
* El decorador mantiene una referencia del objeto actual que va a envolver.
* El decorador recibe todas las peticiones del cliente y las redirecciona al objeto base.
* El objeto decorador agrega funcionalidad antes o después de redireccionar al objeto base. Esto asegura que se agregue funcionalidad al objeto dado en tiempo de ejecución sin alterar su estructura.
* A diferencia de la herencia, el uso del decorador evita una explosión de subclases cuando la jerarquía de clases crezca.



* **En caso de una nueva clase en la jerarquía:** solo seria necesario pasar como argumento el objeto de la nueva clase al decorador y usar sus métodos para lograr la extensión de funcionalidad y no requiere generar mas subclases ni afectar la estructura de la clase en cuestión.
* **En caso de agregar un nuevo decorador:** Como el decorador contiene un ejemplar del objeto base (objeto que envuelve) puede redireccionar el mensaje a este antes o después de agregar la funcionalidad. Como el Decorador es una realización de la misma interfaz que el objeto base, cualquier ejemplar del decorador y cualquier de sus subclases pueden ser tratadas como un objeto de esta jerarquía. Dado que un decorador puede contener otro decorador es posible crear nuevos decoradores y funcionalidad al envolver objetos decoradores ya existentes.

