**SINGLETON PATTERN**

**Los posibles casos de uso de este patrón serian.**

* Cuando una clase en su programa debe tener una sola instancia disponible para todos los clientes; por ejemplo, un único objeto de base de datos compartido por diferentes partes del programa.
* Cuando necesita un control más estricto sobre las variables globales.

**BENEFICIOS:**

* Puede estar seguro de que una clase tiene una sola instancia.
* Obtiene un punto de acceso global a esa instancia.
* El objeto singleton se inicializa solo cuando se solicita por primera vez.

**DESVENTAJAS:**

* Viola el principio de responsabilidad única.
* El patrón Singleton puede enmascarar un mal diseño, por ejemplo, cuando los componentes del programa se conocen demasiado entre sí.
* El patrón requiere un tratamiento especial en un entorno multiproceso para que múltiples hilos no creen un objeto singleton varias veces.
* Puede ser difícil hacer pruebas unitarias código del cliente de Singleton porque muchos marcos de prueba dependen de la herencia al producir objetos simulados.