Kenneth Galvez 20079

Yong Bum Park 20117

Singleton

Singleton es un patrón de diseño innovador que garantiza que solo exista un objeto de su tipo y proporciona un único punto de acceso a él para cualquier otro código(Patrones de diseño: Singleton en Java, 2014).

Algunos beneficios del uso de Singleton son:

- Reducir el espacio de nombres: Este modo es una mejora de las variables globales. El nombre ya no está reservado para variables globales, ahora solo existen instancias
- Controle el acceso a una sola instancia, porque la clase Singleton encapsula una sola instancia: Esto le permite controlar cómo y cuándo acceder a él.
- Permite el refinamiento de las operaciones y representación.
- Permite un número variable de instancias: El patron es fácil de configurar para permitir múltiples instancias.

(Singleton | Marco de Desarrollo de la Junta de Andalucía, 2021)

Algunas desventajas del uso de Singletos son:

- Dificultad en la prueba: los singleton son como valores de variables globales. Global y orientado a objetos no son buenos amigos, porque introduce un "estado persistente", lo que significa que el valor siempre se mantiene, lo que dificulta el uso de objetos de reemplazo (simulación) en las pruebas.
- Promover un alto acoplamiento: si algo debe estar alto en la dirección del objeto, es cohesión en lugar de acoplamiento. Singleton se crea directamente desde su propia clase, lo que promueve el uso de métodos privados y estáticos. Esto empareja las clases que las usan juntas y evita el uso adecuado de la inyección de dependencia.
- Limitaciones de la ejecución en paralelo: aunque el objetivo de Singleton es administrar recursos compartidos, esto limita el cuello de botella de las operaciones en paralelo en las aplicaciones y las convierte en operaciones en serie, lo que no se recomienda cuando la demanda es alta.

(Franky Villadiego, 2013)

Creemos que utilizar Singleton en esta hoja de trabajo esta bien, tal vez no lo mas optimo, pero si cumple con el objetivo. Todas las implementaciones que utilizamos en nuestro programa requieren de una instancia que se realiza en otra clase para poder realizar las operaciones por lo que sí vemos adecuado su uso.

Bibliografía

Patrones de diseño: Singleton en Java. (2014). Refactoring.guru. https://refactoring.guru/es/design-patterns/singleton/java/example

Franky Villadiego. (2013). *El objeto único [Patrón Singleton]*. Eljaviador.com. http://eljaviador.com/el-objeto-unico-patron-singleton.html

Singleton | Marco de Desarrollo de la Junta de Andalucía. (2021). Juntadeandalucia.es. http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/202