Singleton es un patrón de diseño creacional que nos permite asegurarnos de que una clase tenga una única instancia, a la vez que proporciona un punto de acceso global a dicha instancia.

Seria tener un solo objeto global para toda la aplicación, que podrá ser llamado desde distintas clases.

Reglas:

1-debemos tener un constructor vacío privado

2-debemos tener un atributo privado estático

3-debemos tener un método estático que retorne la instancia

IMPORTANTE SI SE TRABAJA CON HILOS SICRONIZARLOS, PARA EVITAR PROBLEMAS AL CREAR LOS OBJETOS.

Spring trabaja con este patron y prototype!!!

Ejemplo en código:

Creamos este objeto con el patron singleton

public class DatabaseConnector {  
 private static DatabaseConnector *databaseConnector*;  
 private DatabaseConnector(){  
 System.*out*.println("Creando Objeto");  
 }  
  
 public static synchronized DatabaseConnector getInstance(){  
 if (*databaseConnector*==null){  
 *databaseConnector*=new DatabaseConnector();  
 }  
 return *databaseConnector*;  
 }  
  
 public void connectDbase(){  
 System.*out*.println("Conectando -> Base de Datos = " + *databaseConnector*);  
 }  
 public void disconnectDbase(){  
 System.*out*.println("Desconectando -> Base de Datos = " + *databaseConnector*);  
 }  
  
}

public class Main {  
 public static void main(String[] args) {  
 DatabaseConnector databaseConnector= DatabaseConnector.*getInstance*();  
 System.*out*.println("databaseConnector= "+ databaseConnector);  
  
 DatabaseConnector databaseConnector2= DatabaseConnector.*getInstance*();  
 System.*out*.println("databaseConnector2= "+ databaseConnector2);  
 System.*out*.println("/n");  
 databaseConnector.connectDbase();  
 System.*out*.println("/n");  
 databaseConnector2.connectDbase();  
 }  
}