**Deep Dive “Bean Scope”**

singleton (default) – one instance per Spring Context

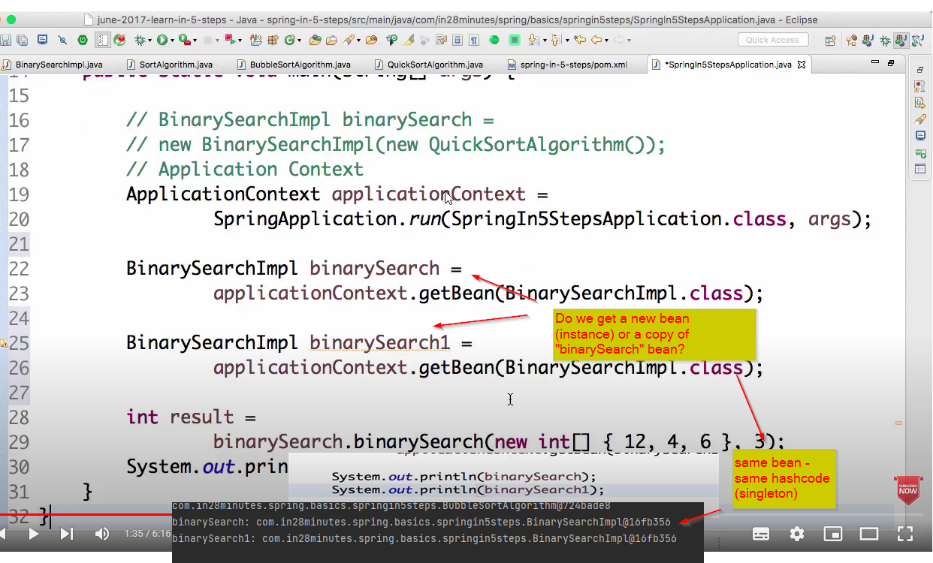
prototype – new bean whenever requested

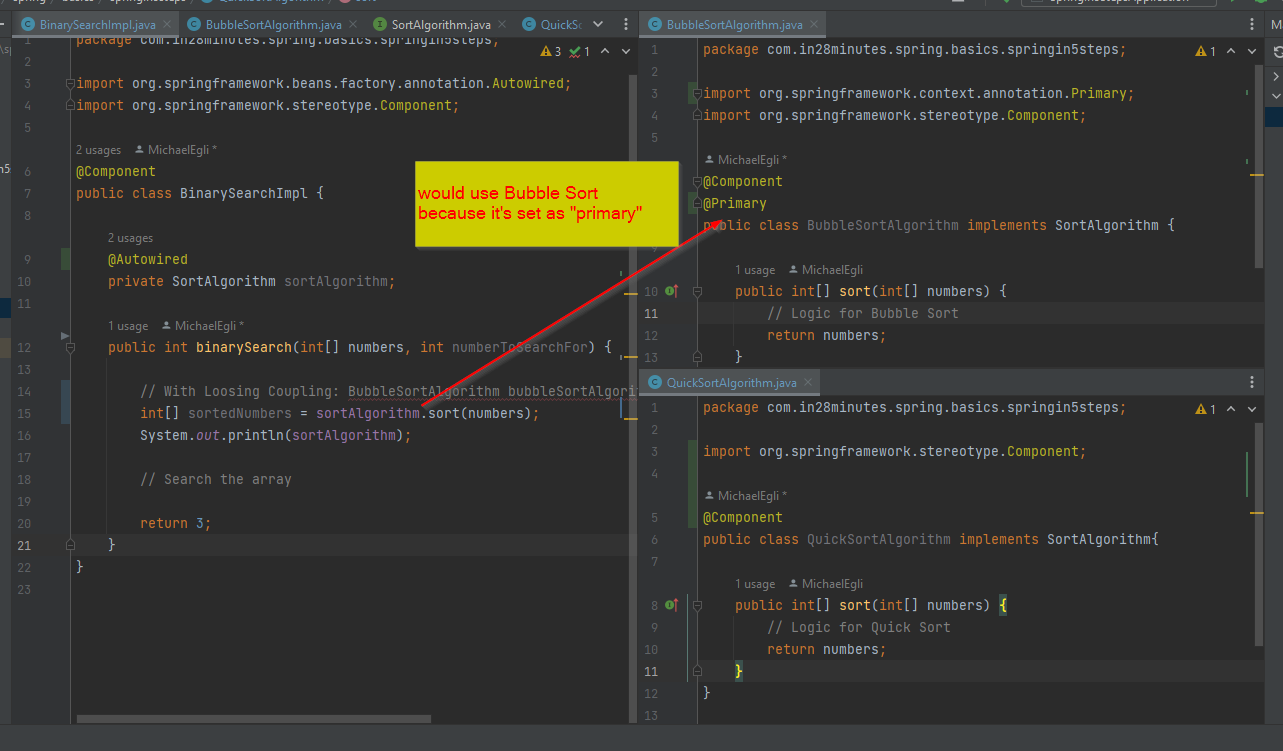
request – one bean per http request

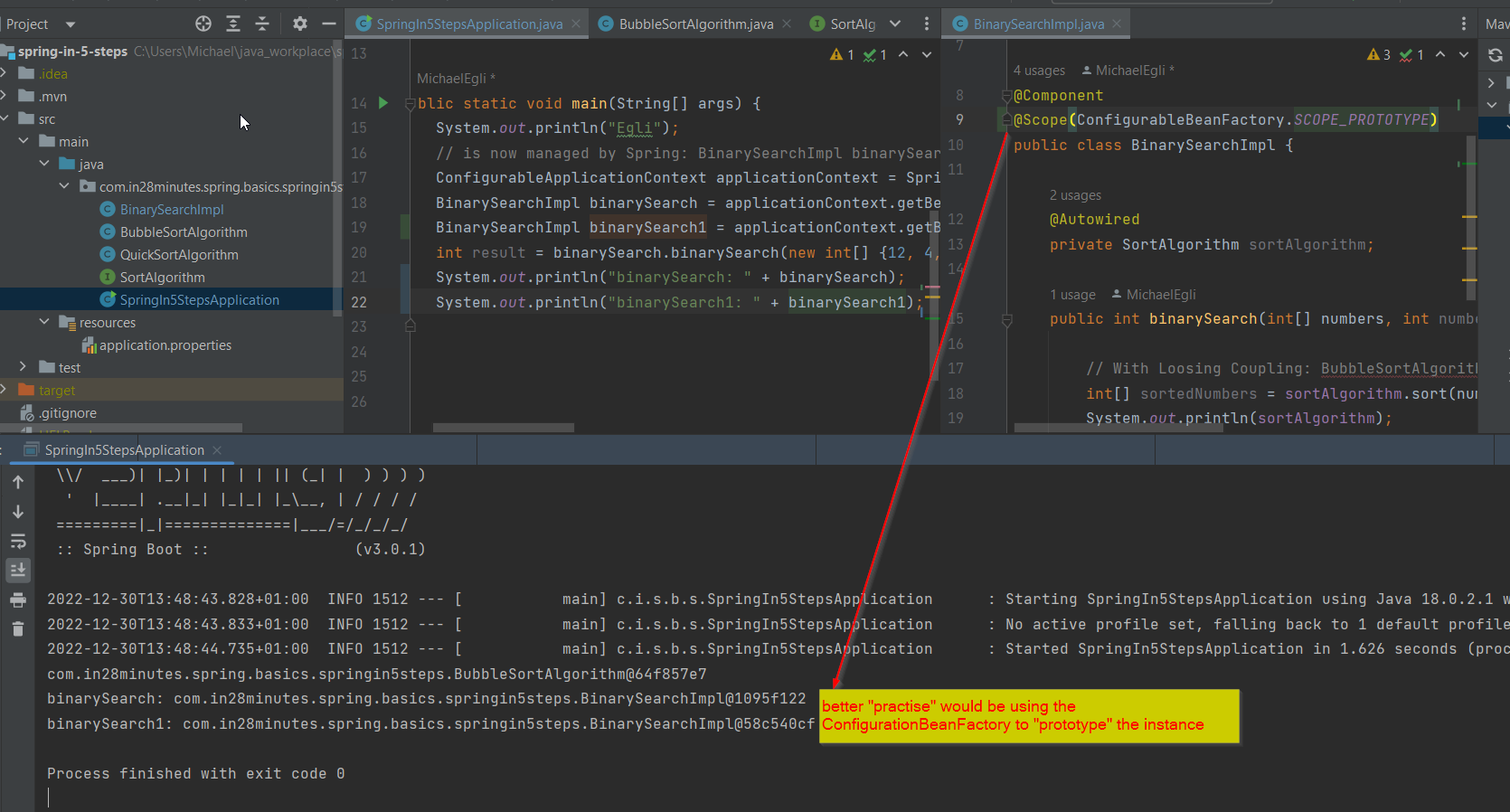
session – one bean per http (user) session

The life-cycle of a bean is manager by a container.

Singleton vs Prototype



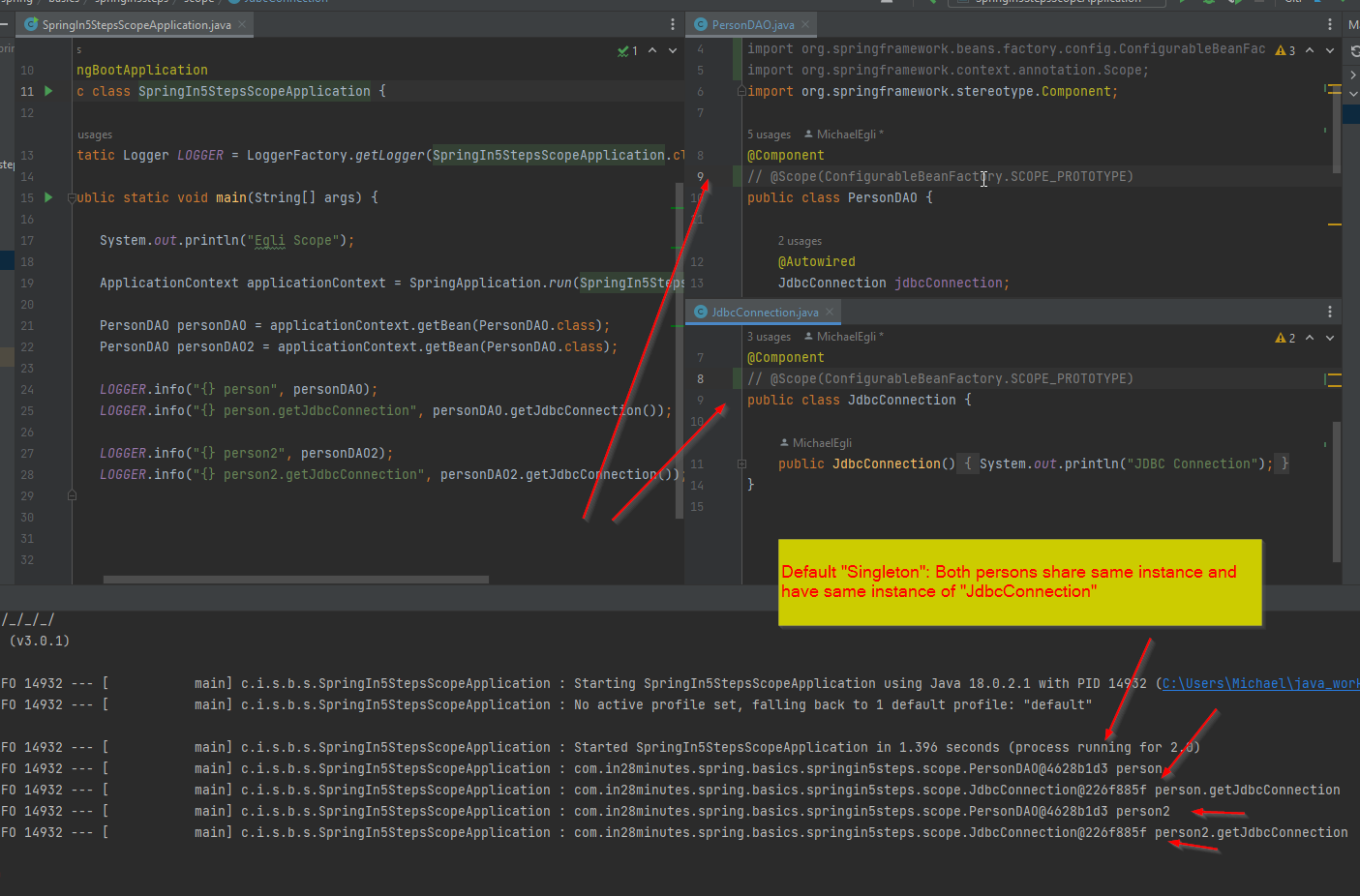




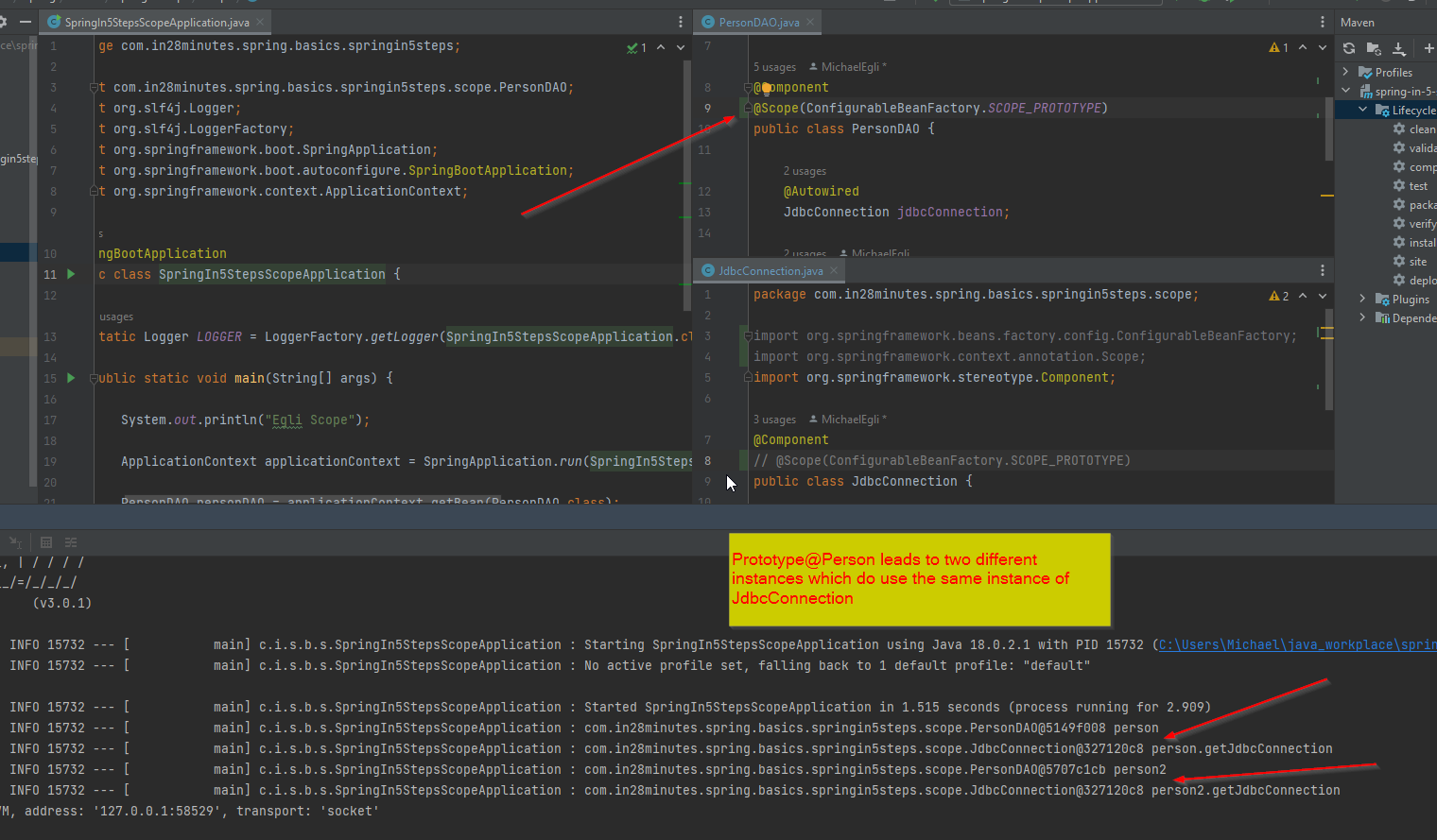
What happens if you create an object that does inject other objects (dependencies)?

f.e. You have an object “Person” which does inject a “JdbcConnection”:

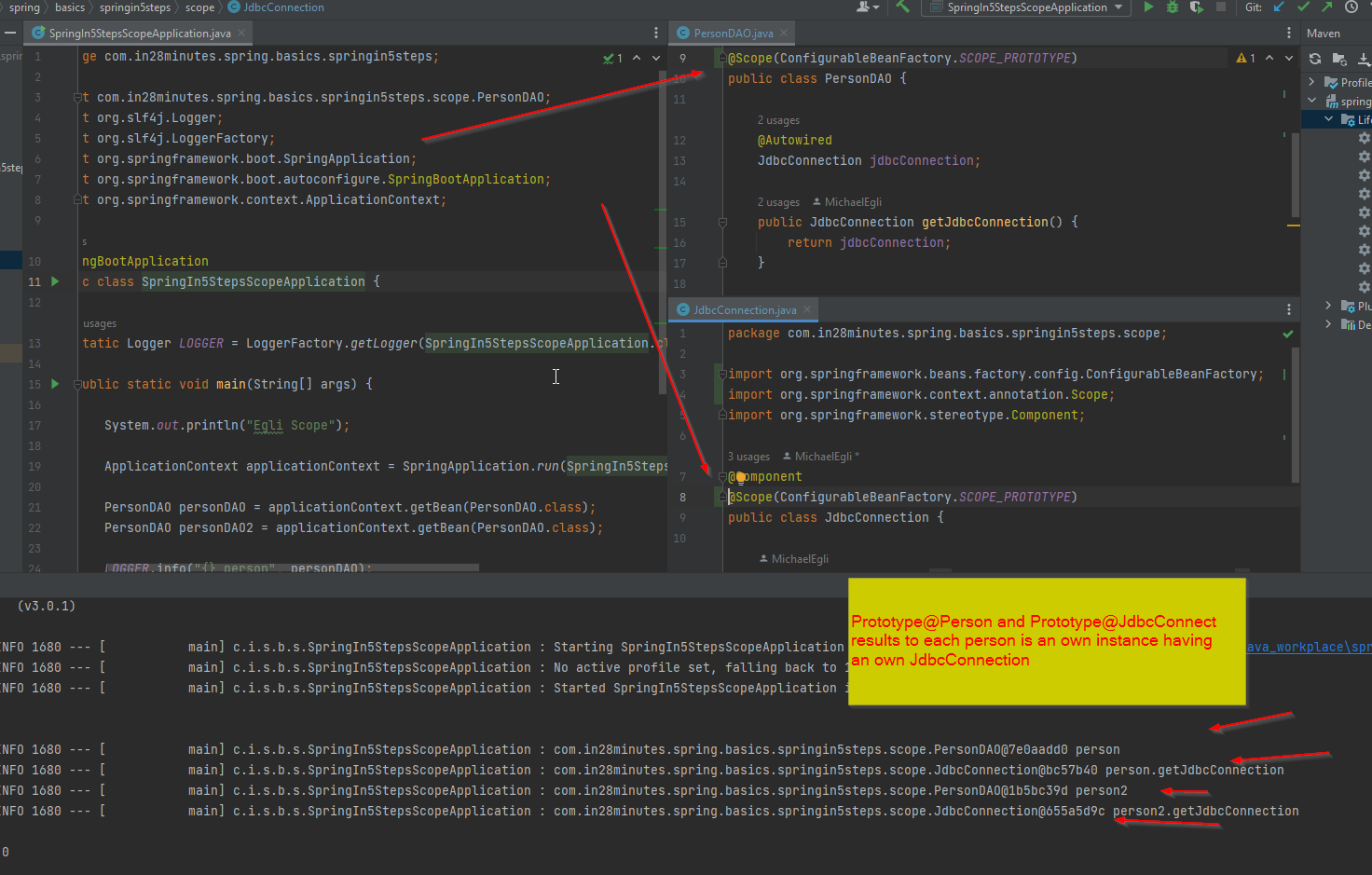
Case 1 “Default uses Singleton”



Case 2 “Prototype@Person”

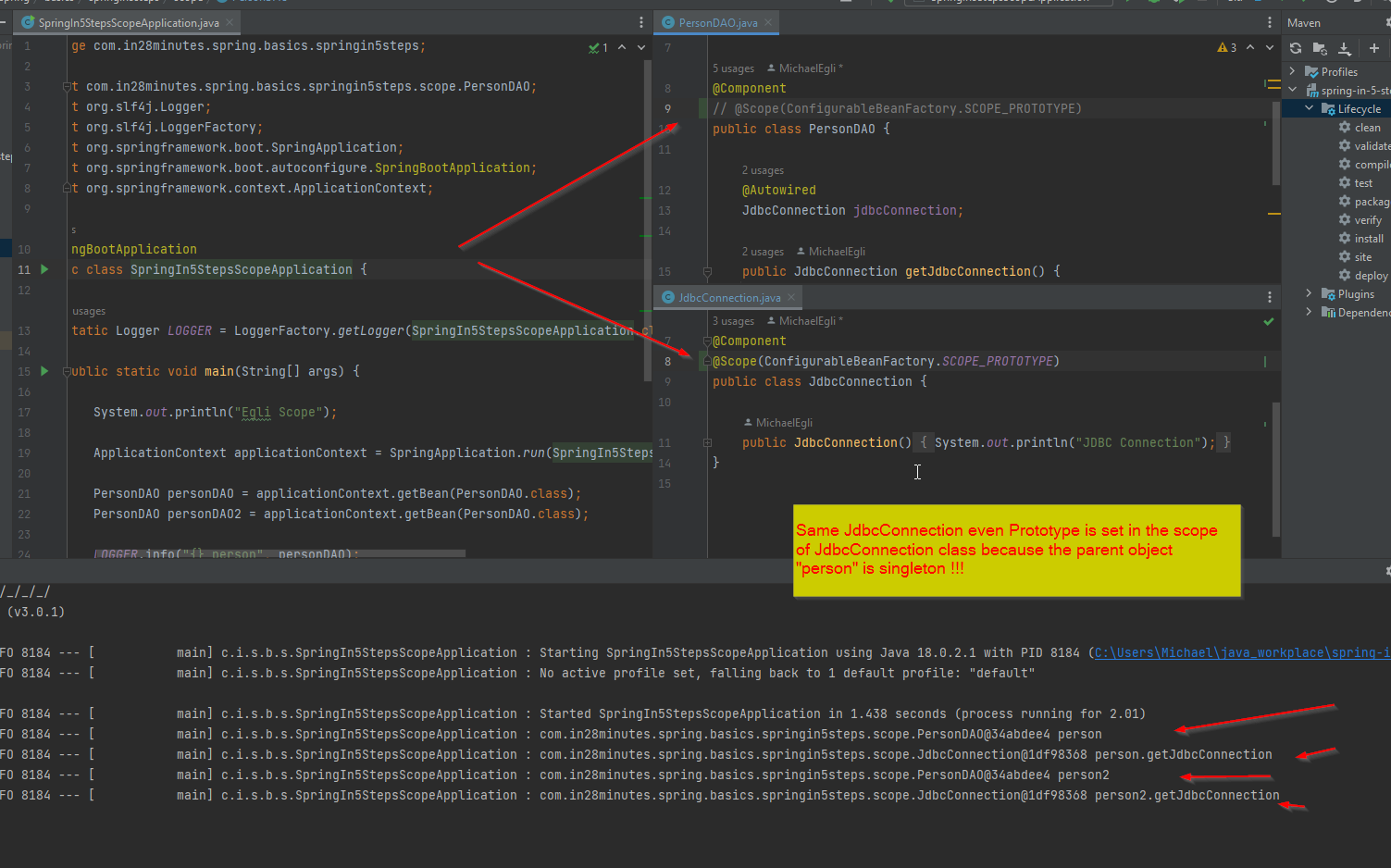


Case 3 “Protoype@Person and Prototype@JdbcConnection”



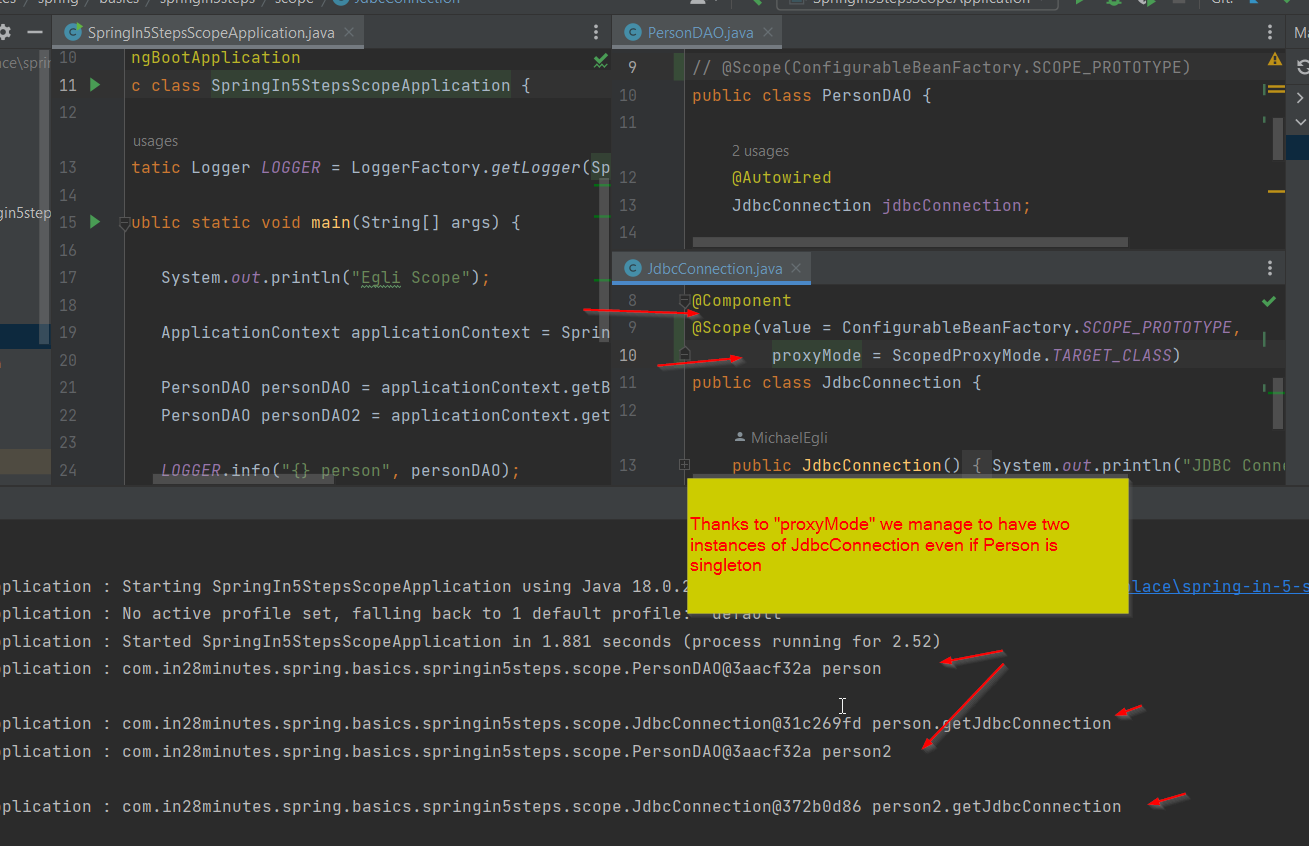
Case 4 “Prototype@JdbcConnection”

Now this is the important case to understand. Due the fact the person is singleton, they will have the same JdbcConnection even “prototype” is set on the “JdbcConnection” class.



Case 5 “Each JdbcConnection should be a proper instance even Person is singleton”

To solve this problem (of Case 4) we must configure a proxy.



Important: Each time you call “getJdbcConnection()” of a person (even als Logger.info(…)), a new instance of JdbcConnection would be created (therefore a singleton person could have multiple JdbcConnections).

Singleton – additional notes

Scope for singleton is typically one instance of a JVM. In Spring the scope of a singleton is one instance in the Spring Application Context (it’s possible to run multiple Application Contexts in one JVM).