Final class & Final method in Spring

=

Class annotated with `@Configuration` cannot be final because Spring will use CGLIB to create a proxy for `@Configuration` class. CGLIB creates subclass for each class that is supposed to be proxied, however since final class cannot have subclass CGLIB will fail. This is also a reason why methods cannot be final, Spring needs to override methods from parent class for proxy to work correctly, however final method cannot be overridden, having such a method will make CGLIB fail.

If `@Configuration` class will be final or will have final method, Spring will throw `BeanDefinitionParsingException`.

Spring supports Singleton beans in `@Configuration` class by creating CGLIB proxy that intercepts calls to the method. Before method is executed from the proxied class, proxy intercept a call and checks if instance of the bean already exists, if instance of the bean exists, then call to method is not allowed and already existing instance is returned, if instance does not exists, then call is allowed, bean is created and instance is returned and saved for future reuse. To make method call interception CGLIB proxy needs to create subclass and also needs to override methods.

Easiest way to observe that calls to original `@Configuration` class are proxied is with usage of debugger or by printing stacktrace. When looking at stacktrace you will notice that class which serves beans is not original class written by you but it is different class, which name contains `$$EnhancerBySpringCGLIB`.