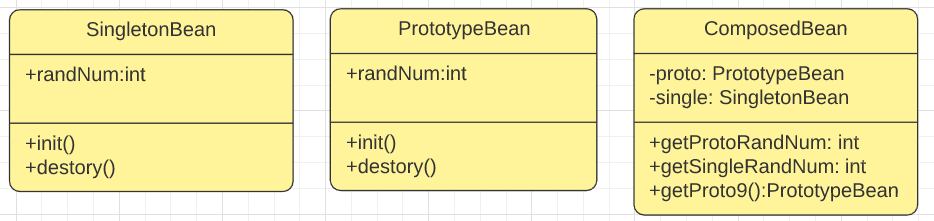
Exercise No. 5

Target: Beans Scopes, Beans lifecycle and Beans interceptor.

* + There are three phases in Spring bean's life cycle:
  + Created/Ready - A creation of a Spring bean can be a complex action.
  + Used - Beans are the performers/actors of the application.
  + Destroyed - At Spring container shutdown, beans are destroyed.
  + Spring bean factory is responsible for managing the life cycle of beans created through spring container.
  + The life cycle of beans consists of call back methods which can be categorized broadly in two groups:
  + Post initialization call back methods
  + Pre destruction call back methods
  + Spring has 2 types of Beans scopes, Singleton- the Bean will have 1 instance in the current Spring Core container, prototype – Bean will have more than 1 bean in the current Spring Core Container
  + Beans Lifecycle & Interceptors

postProcessBeforeInitialization() , postProcessAfterInitialization()

1. Open template for exercise 5 by using the following link: <https://gitlab.corp.amdocs.com/DAWODK/springbootframework/tree/master/Templates/Exercise5>
2. Open the package “beans” under - src/main/java and Create the following Classes under beans package



SingletonBean:

* Add the @Component @Scope("*singleton*").
* Add the annotation @PostConstruct to the method init ().
  + System.***out***.println("Inside init method for Singleton");
* Add the annotation @PreDestroy to the method destroy ().
  + System.***out***.println("Inside destroy method for Singleton");
* Initiate the randNum with Random Number between 0 -100.

PrototypeBean:

* Add the @Component("proto") @Scope("prototype").
* Add the annotation @PostConstruct to the method init ().
* Add the annotation @PreDestroy to the method destroy ().
* Initiate the randNum with Random Number between 0 -100.

ComposedBean:

* Add @Autowired to the single class data member
* Add @Autowired and @Qualifier to the single class member
* getProtoRandNum() will invoke the proto.randnum.
* getSingleRandNum will invoke the single.randnum.
* getProto9 will return instance of proto with specific random number 9.

1. Runner Class
   * Open SpringMain Class
   * Define the main Class as the configuration class by the annotation @Configuration
   * Define the scanned packages “the packages which will include the Classes with the stereotype annotation with the annotation @ComponentScan
   * Inside the main, create Spring Core Container context by Using the *AnnotationConfigApplicationContext* option with the annotation metadata.
   * Create 2 instances per class (Beans) from the PrototypeBean and SingltonBean.
   * Create 2 instances from the ComposedBean
   * Print all random numbers values from the instances that you create.
   * Create instance from the proto9 Bean and print the random number from this instance.
2. Open the package interceptor

* Add the package interceptor to the annotation @ComponentScan in the main Runner
* Complete the implementations of the *postProcessBeforeInitialization* and *postProcessAfterInitialization*
  + 1. *postProcessBeforeInitialization: in case of prototypeBean instance and BeanName.eqauls to proto* 
       1. print the instance name
       2. check if the random number is Odd, if yes add it by 1.
       3. return the Bean.
    2. *postProcessBeforeInitialization: in case of prototypeBean instance and BeanName.eqauls to proto* 
       1. print Final randnum value = b.randnum.
       2. print the Bean name.

1. add the following to the SpringMain

PrototypeBean p4=ctx.getBean("proto",PrototypeBean.**class**);

Please review below link for full solution:

<https://gitlab.corp.amdocs.com/DAWODK/springbootframework/tree/master/FullSolutions/Exercise5>

Thank you