@PostConstruct and @PreDestroy

Custom init method and custom destroy method previously we have done using **xml file as shown below**

- First we have Inserted **myStartup() and myClosing()** in Audi class

```
public class Audi implements Car
{
	void myStartUp(){
		System.out.println("Bean Created");
	}

	void myClosing(){
		System.out.println("Bean Destroyed");
	}

}
```

- inserted **init-method and destroy-method** is the attribute of spring <bean> tag in applicationContext.

```
<bean id ="audi" class="com.tapacad.spring.Audi"
                        init-method="myStartUp"
                        destroy-method="myClosing">
	<property name="rocketEngine" ref = "engine"></property>
	<property name="colour" value = "${colour}"></property>
	<property name="price" value = "${price}"></property>
</bean>
```

Now let's achieve the same thing using annotation
Here @PostConstruct is used to represent init-method and @PreDestroy to represent destroy method, this we need to mention for myStartUp() and myClosing() available in Audi class

@PostConstruct and @PreDestroy annotations will not be available after jdk 9 to get that download jar file using the link below

https://mvnrepository.com/artifact/javax.annotation/javax.annotation-api/1.2

Now download the jar file and add the jar file into the project

Audi.java

```
@Component("audi")
public class Audi implements Car
{

	@PostConstruct
	void myStartUp(){
		System.out.println("Bean Created");
	}

	@PreDestroy
	void myClosing(){
		System.out.println("Bean Destroyed");
	}

}
```

MyApp9.java

```
package com.tapacad.spring;
import
org.springframework.context.support.ClassPathXmlApplicationContext;

public class MyApp9 {
	public static void main(String[] args) {
//		Load Application context
		ClassPathXmlApplicationContext context =
```

```
                        new
ClassPathXmlApplicationContext("annotateApplicationContext.xml");

//              Get bean
                Car car1 = context.getBean("audi", Car.class);

//              print car1 and car2 object reference
                System.out.println(car1);

//              close context
                context.close();

        }

}
```

Output:

```
Bean Created
com.tapacad.spring.Audi@771a660
Bean Destroyed
```

If you observe from the above output myStartUp() executes when bean is instantiated and myDestroy() before bean gets destroyed. **Here the scope of bean is singleton**