

# HOL : Spring Object Lifecycle

## Task 1. Lab

### 1. In Package Explorer > right-click > New > Java Project

1)Project Name : SpringLifecycle

2)JRE

-Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

3)Uncheck [Create module-info.java file]

4)Next

5)Finish

### 2. src > right-click > New > Package

1)Name : com.example

2)Finish

### 3. POJO 객체 생성

1)src/com.example > New > Class

2)Name : Student

```
package com.example;
```

```
import java.util.List;
```

```
public class Student {  
    private String name;  
    private int age;  
    private List<String> hobbies;  
    private double height;  
    private double weight;  
}
```

### 4. Java Project를 Spring Project로 변환

1)SpringLifecycle Project > right-click > Configure > Convert to Maven Project

-Project : /SpringLifecycle

-Group Id : SpringLifecycle

-Artifact Id : SpringLifecycle

-version : 0.0.1-SNAPSHOT

-Packaging : jar

-Finish

2)SpringLifecycle Project > right-click > Spring > Add Spring Project Nature

3)pom.xml 파일에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
```

```
<dependencies>
```

```
  <dependency>
```

```
    <groupId>org.springframework</groupId>
```

```
    <artifactId>spring-context</artifactId>
```

```
    <version>5.3.10</version>
```

```
  </dependency>
```

```
</dependencies>
```

4)pom.xml > right-click > Run As > Maven install

[INFO] BUILD SUCCESS 확인

## 5. Lombok library 추가

1) <https://mvnrepository.com/>에서 'lombok'으로 검색

2) 'Project Lombok' click

3) 1.18.20 click

4) dependency copy해서 pom.xml에 붙여넣기

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.10</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.20</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```

5) pom.xml > right-click > Run As > Maven install

[INFO] BUILD SUCCESS 확인

## 6. Student.java 수정

1) Student.java

```
package com.example;

import java.util.List;

import lombok.NonNull;
import lombok.RequiredArgsConstructor;
import lombok.Setter;
import lombok.ToString;

@RequiredArgsConstructor
@ToString
public class Student {
    @NonNull private String name;
    @NonNull private Integer age;
    @NonNull private List<String> hobbies;
    @Setter private double height;
    @Setter private double weight;
}
```

## 7. SpringLifecycle/resources folder 생성

1) SpringLifecycle project > right-click > New > Source Folder

2) Folder name : resources

3) Finish

## 8. Bean Configuration XML 작성

1)SpringLifecycle/resources > right-click > New > Spring Bean Configuration File

2)File name : applicationContext.xml

3)Namespaces에서 util check할 것

3)Finish

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<beans xmlns="http://www.springframework.org/schema/beans"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:util="http://www.springframework.org/schema/util"
```

```
xsi:schemaLocation="http://www.springframework.org/schema/beans
```

```
http://www.springframework.org/schema/beans/spring-beans.xsd">
```

```
<bean id="student1" class="com.example.Student">
```

```
<constructor-arg value="백두산" />
```

```
<constructor-arg value="25" />
```

```
<constructor-arg>
```

```
<util:list>
```

```
<value>독서</value>
```

```
<value>영화감상</value>
```

```
<value>요리</value>
```

```
</util:list>
```

```
</constructor-arg>
```

```
<property name="height" value="165" />
```

```
<property name="weight">
```

```
<value>45</value>
```

```
</property>
```

```
</bean>
```

```
</beans>
```

## 9. com.example.MainClass.java

```
package com.example;
```

```
import org.springframework.context.support.GenericXmlApplicationContext;
```

```
public class MainClass {
```

```
    public static void main(String[] args) {
```

```
        GenericXmlApplicationContext context = new
```

```
        GenericXmlApplicationContext();
```

```
        context.load("classpath:applicationContext.xml");
```

```
        context.refresh();
```

```
        Student student1 = context.getBean("student1", Student.class);
```

```
        System.out.println(student1);
```

```
        context.close();
```

```
    }
```

```
}
```

## 10. 실행

1)MainClass > right-click > Run As > Java Application

Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0, weight=45.0)

```

166
167
168 11. Java Annotation 방식 사용하기
169     1)src/com.example > right-click > New > Class
170     2)Name : ApplicationConfig
171
172         package com.example;
173
174         import java.util.Arrays;
175         import java.util.List;
176
177         import org.springframework.context.annotation.Bean;
178         import org.springframework.context.annotation.Configuration;
179
180         @Configuration
181         public class ApplicationConfig {
182             @Bean
183             public Student student1() {
184                 List<String> list = Arrays.asList("독서", "영화감상", "요리");
185                 Student student = new Student("백두산", 25, list);
186                 student.setHeight(165.0);
187                 student.setWeight(45.0);
188                 return student;
189             }
190         }
191
192     3)com.example.MainClass2 생성
193     -src/com.example > right-click > New > Class
194     -Name : MainClass2
195
196         package com.example;
197
198         import
199         org.springframework.context.annotation.AnnotationConfigApplicationContext
200         ;
201
202         public class MainClass2 {
203             public static void main(String[] args) {
204                 AnnotationConfigApplicationContext ctx = new
205                 AnnotationConfigApplicationContext(ApplicationConfig.class);
206                 Student student1 = ctx.getBean("student1", Student.class);
207                 System.out.println(student1);
208
209                 Student student2 = ctx.getBean("student1", Student.class);
210                 System.out.println(student1 == student2);
211                 ctx.close();
212             }
213         }
214
215     4)실행 결과
216     Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,
217     weight=45.0)
218     true
219
220 12. @Component 방식으로 Student 객체 변경

```

```

218 1)Student.java
219
220     package com.example;
221
222     import java.util.List;
223
224     import org.springframework.beans.factory.annotation.Value;
225     import org.springframework.stereotype.Component;
226
227     import lombok.Getter;
228     import lombok.ToString;
229
230     @ToString
231     @Getter
232     @Component(value = "student1")
233     public class Student {
234         @Value("한라산")
235         private String name;
236         @Value("35")
237         private Integer age;
238         @Value("등산, 게임, 독서")
239         private List<String> hobbies;
240         @Value("162.5")
241         private double height;
242         @Value("49.2")
243         private double weight;
244     }
245
246 2)com.example.ApplicationConfig 변경
247
248     package com.example;
249
250     import org.springframework.context.annotation.ComponentScan;
251     import org.springframework.context.annotation.Configuration;
252
253     @Configuration
254     @ComponentScan(basePackages = {"com.example"})
255     public class ApplicationConfig {}
256
257 3)resources/applicationContext.xml 삭제
258 4)com.example.MainClass2 실행 결과
259
260     Student(name=한라산, age=35, hobbies=[등산, 게임, 독서], height=162.5,
261     weight=49.2)
262     true
263
264 13. JUnit 5를 사용한 DI test class(JUnit5Test.java) 작성
265     1)src > right-click > New > Package
266     2)Name : com.example.test
267     3)Finish
268
269     4)com.example.test > right-click > New > JUnit Test Case
270     5)Select [New JUnit Jupiter test]
271     6)Name : JUnit5Test
272     7)Finish

```

```

8)Select [Perform the following action: Add Juni 5 library to the build path
9)OK

package com.example.test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.example.ApplicationConfig;
import com.example.Student;

class JUnit5Test {
    private AnnotationConfigApplicationContext context;

    @BeforeEach
    public void init() {
        this.context = new
        AnnotationConfigApplicationContext(ApplicationConfig.class);
    }

    @Test
    public void test1() {
        Student student = this.context.getBean("student1", Student.class);
        assertEquals("한라산", student.getName());
    }
}

```

4)right-click > Run As > JUnit Test

5)결과 -> Junit View에 초록색 bar

## Task 2. Lab

1. In Package Explorer > right-click > New > Java Project

1)Project Name : SpringLifecycle1

2)JRE

-Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

3)Uncheck [Create module-info.java file]

4)Next

5)Finish

2. src > right-click > New > Package

1)Package name : com.example

3. POJO 객체 생성

1)com.example > right-click > New > Class

2)Name : Student.java

```

package com.example;

```

```
327     public class Student{
328         private String name;
329         private int age;
330     }
331
332
```

#### 333 4. Java Project를 Spring Project로 변환

334 1)SpringLifecycle1 Project > right-click > Configure > Convert to Maven Project

335 -Project : /SpringLifecycle1

336 -Group Id : SpringLifecycle1

337 -Artifact Id : SpringLifecycle1

338 -version : 0.0.1-SNAPSHOT

339 -Packaging : jar

340 -Finish

342 2)SpringLifecycle1 Project > right-click > Spring > Add Spring Project Nature

344 3)pom.xml 파일에 Spring Context Dependency 추가하기

345 <version>0.0.1-SNAPSHOT</version>

346 <dependencies>

347 <dependency>

348 <groupId>org.springframework</groupId>

349 <artifactId>spring-context</artifactId>

350 <version>5.3.10</version>

351 </dependency>

352 </dependencies>

354 4)pom.xml > right-click > Run As > Maven install

355 [INFO] BUILD SUCCESS 확인

#### 358 5. Lombok library 추가

359 1)<https://mvnrepository.com/>에서 'lombok'으로 검색

360 2)'Project Lombok' click

361 3)1.18.20 click

362 4)dependency copy해서 pom.xml에 붙여넣기

364 <dependencies>

365 <dependency>

366 <groupId>org.springframework</groupId>

367 <artifactId>spring-context</artifactId>

368 <version>5.3.10</version>

369 </dependency>

370 <!-- <https://mvnrepository.com/artifact/org.projectlombok/lombok> -->

371 <dependency>

372 <groupId>org.projectlombok</groupId>

373 <artifactId>lombok</artifactId>

374 <version>1.18.20</version>

375 <scope>provided</scope>

376 </dependency>

377 </dependencies>

379 5)pom.xml > right-click > Run As > Maven install

380 [INFO] BUILD SUCCESS 확인

383 6. Student.java lombok Annotation 붙이고, InitializingBean, DisposableBean  
interface 구현하기

```
384  
385 package com.example;  
386  
387 import org.springframework.beans.factory.DisposableBean;  
388 import org.springframework.beans.factory.InitializingBean;  
389  
390 import lombok.AllArgsConstructor;  
391 import lombok.ToString;  
392  
393 @AllArgsConstructor  
394 @ToString  
395 public class Student implements InitializingBean, DisposableBean{  
396     private String name;  
397     private int age;  
398  
399     @Override  
400     public void destroy() throws Exception {  
401         System.out.println("방금 Bean이 소멸됐습니다.");  
402     }  
403     @Override  
404     public void afterPropertiesSet() throws Exception {  
405         System.out.println("방금 Bean이 생성됐습니다.");  
406     }  
407 }  
408  
409
```

410 7. SpringLifecycle1/resources folder 생성

- 411 1)SpringLifecycle1 project > right-click > New > Source Folder
- 412 2)Folder name : resources
- 413 3)Finish

414

415

416 8. Bean Configuration XML 작성

- 417 1)SpringLifecycle1/resources > right-click > New > Spring Bean Configuration File
- 418 2)File name : applicationContext.xml
- 419 3)Finish

420

```
421 <?xml version="1.0" encoding="UTF-8"?>  
422 <beans xmlns="http://www.springframework.org/schema/beans"  
423     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
424     xsi:schemaLocation="http://www.springframework.org/schema/beans  
http://www.springframework.org/schema/beans/spring-beans.xsd">  
425  
426     <bean id="student" class="com.example.Student">  
427         <constructor-arg value="백두산" />  
428         <constructor-arg value="25" />  
429     </bean>  
430  
431 </beans>  
432  
433
```

434 9. MainClass 작성

- 435 1)com.example > right-click > New > Class
- 436 2)Name : MainClass



### 3)Finish

```
package com.example;

import org.springframework.context.support.GenericXmlApplicationContext;

public class MainClass {
    public static void main(String[] args) {
        GenericXmlApplicationContext context = new
            GenericXmlApplicationContext();
        context.load("classpath:applicationContext.xml");
        context.refresh();

        Student student = context.getBean("student", Student.class);
        System.out.println(student);
        context.close();
    }
}
```

### 10. 실행

1)MainClass > right-click > Run As > Java Application

방금 Bean이 생성되었습니다.

Student (name=백두산, age=25)

방금 Bean이 소멸되었습니다.

### 11. @PostConstruct, @PreDestroy 이용하기

1)<https://mvnrepository.com>에서 'jaxax annotation'으로 검색

2)[Javax Annoation API] click

3)1.3.2 click

4)dependency copy하여 pom.xml에 paste

```
<!--
https://mvnrepository.com/artifact/javax.annotation/javax.annotation-api -->
<dependency>
    <groupId>javax.annotation</groupId>
    <artifactId>javax.annotation-api</artifactId>
    <version>1.3.2</version>
</dependency>
```

5)pom.xml > right-click > Run As > Maven install

[INFO] BUILD SUCCESS 확인

6)Student2 class 생성하기

-com.example > right-click > New > Class

-Name : Student2

```
package com.example;

import javax.annotation.PostConstruct;
import javax.annotation.PreDestroy;

import lombok.AllArgsConstructor;
import lombok.ToString;
```

```

491 @AllArgsConstructor
492 @ToString
493 public class Student2 {
494     private String name;
495     private int age;
496
497     @PostConstruct // Bean이 생성단계에서 해야할 일 기술
498     public void initTest() {
499         System.out.println("방금 객체가 생성됐습니다.");
500     }
501
502     @PreDestroy // Bean이 소멸할 때 해야할 일 기술
503     public void destroyTest() {
504         System.out.println("방금 객체가 소멸됐습니다.");
505     }
506 }
507
508

```

## 12. resources/applicationContext.xml 수정하기

1)Namespaces Tab click

2)[context] check

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

```

```
<!-- 첫번째 방법 -->
```

```
<context:annotation-config/>
```

```
<bean id="student2" class="com.example.Student2">
```

```
    <constructor-arg value="백두산" />
```

```
    <constructor-arg value="25" />
```

```
</bean>
```

```
<!-- 두번째 방법-->
```

```
<bean
```

```
class="org.springframework.context.annotation.CommonAnnotationBeanPostPr
ocessor" />
```

```
<bean id="student2" class="com.example.Student2">
```

```
    <constructor-arg value="백두산" />
```

```
    <constructor-arg value="25" />
```

```
</bean>
```

```
<!-- 세번째 방법 -->
```

```

<bean id="student2" class="com.example.Student2" init-method="initTest"
destroy-method="destroyTest">

```

```
    <constructor-arg value="백두산" />
```

```
    <constructor-arg value="25" />
```

```
</bean>
```

```
</beans>
```

## 13. MainClass 수정

1)com.example.MainClass.java

```

543 package com.example;
544
545 import org.springframework.context.support.GenericXmlApplicationContext;
546
547 public class MainClass {
548     public static void main(String[] args) {
549         GenericXmlApplicationContext context = new
            GenericXmlApplicationContext();
550         context.load("classpath:applicationContext.xml");
551         context.refresh();
552
553         Student2 student2 = context.getBean("student2", Student2.class);
554         System.out.println(student2);
555         context.close();
556     }
557 }

```

## 2)실행

-MainClass > right-click > Run As > Java Application

방금 객체가 생성됐습니다.

Student2(name=백두산, age=25)

방금 객체가 소멸됐습니다.

## Task 3. Lab

1. In Package Explorer > right-click > New > Java Project

1)Project Name : SpringScopeDemo

2)JRE

-Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

3)Uncheck [Create module-info.java file]

4)Next

5)Finish

2. src > right-click > New > Package

1)Name : com.example

2)Finish

3. com.example.Student class 생성

1)com.example > right-click > New > Class

2)Name : Student

3)Finish

```
package com.example;
```

```
public class Student{
    private String name;
    private int age;
}
```

4. Java Project를 Spring Project로 변환

1)SpringScopeDemo Project > right-click > Configure > Convert to Maven Project

-Project : /SpringScopeDemo

```

598 -Group Id : SpringScopeDemo
599 -Artifact Id : SpringScopeDemo
600 -version : 0.0.1-SNAPSHOT
601 -Packaging : jar
602 -Finish
603
604 2)SpringScopeDemo Project > right-click > Spring > Add Spring Project Nature
605
606 3)pom.xml 파일에 Spring Context Dependency 추가하기
607 <version>0.0.1-SNAPSHOT</version>
608 <dependencies>
609 <dependency>
610 <groupId>org.springframework</groupId>
611 <artifactId>spring-context</artifactId>
612 <version>5.3.10</version>
613 </dependency>
614 </dependencies>
615
616 4)pom.xml > right-click > Run As > Maven install
617 [INFO] BUILD SUCCESS 확인
618
619
620 5. Lombok library 추가
621 1)https://mvnrepository.com/에서 'lombok'으로 검색
622 2)'Project Lombok' click
623 3)1.18.20 click
624 4)dependency copy해서 pom.xml에 붙여넣기
625
626 <dependencies>
627 <dependency>
628 <groupId>org.springframework</groupId>
629 <artifactId>spring-context</artifactId>
630 <version>5.3.10</version>
631 </dependency>
632 <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
633 <dependency>
634 <groupId>org.projectlombok</groupId>
635 <artifactId>lombok</artifactId>
636 <version>1.18.20</version>
637 <scope>provided</scope>
638 </dependency>
639 </dependencies>
640
641 5)pom.xml > right-click > Run As > Maven install
642 [INFO] BUILD SUCCESS 확인
643
644
645 6. Student.java lombok Annotation 붙이기
646
647 package com.example;
648
649 import lombok.AllArgsConstructor;
650 import lombok.Setter;
651 import lombok.ToString;
652
653 @AllArgsConstructor

```

```

654 @Setter
655 @ToString
656 public class Student {
657     private String name;
658     private int age;
659 }
660
661
662 7. SpringLifecycle1/resources folder 생성
663 1)SpringLifecycle1 project > right-click > New > Source Folder
664 2)Folder name : resources
665 3)Finish
666
667
668 8. Bean Configuration XML 작성
669 1)SpringLifecycle1/resources > right-click > New > Spring Bean Configuration File
670 2)File name : applicationContext.xml
671 3)Finish
672
673     <?xml version="1.0" encoding="UTF-8"?>
674     <beans xmlns="http://www.springframework.org/schema/beans"
675           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
676           xsi:schemaLocation="http://www.springframework.org/schema/beans
677                               http://www.springframework.org/schema/beans/spring-beans.xsd">
678
679         <bean id="student" class="com.example.Student" scope="singleton">
680             <constructor-arg value="백두산" />
681             <constructor-arg value="25" />
682         </bean>
683     </beans>
684
685
686 9. MainClass 생성하기
687 1)com.example > right-click > New > Class
688 2)Name : MainClass
689
690     package com.example;
691
692     import org.springframework.context.support.AbstractApplicationContext;
693     import org.springframework.context.support.GenericXmlApplicationContext;
694
695     public class MainClass {
696         public static void main(String[] args) {
697             AbstractApplicationContext context = new
698                 GenericXmlApplicationContext("classpath:applicationContext.xml");
699
700             Student student = context.getBean("student", Student.class);
701             System.out.println(student);
702             System.out.println("-----");
703
704             Student student1 = context.getBean("student", Student.class);
705             student1.setName("북한산");
706             student1.setAge(55);
707             System.out.println(student1);
708             System.out.println("-----");

```

```

708
709         if(student.equals(student1)) System.out.println("Equals"); //Print Equals
710         else System.out.println("Different");
711         context.close();
712     }
713 }
714
715

```

```

716 10. Java Application 실행 결과
717     Student [name=백두산, age=25]
718     -----
719     Student [name=북한산, age=55]
720     -----
721     Equals
722
723

```

```

724 11. ApplicationConfig와 MainClass2 생성하기
725     1)com.example > right-click > New > Class
726     2)Name : ApplicationConfig
727     3)Finish
728

```

```

729     package com.example;
730
731     import org.springframework.context.annotation.Bean;
732     import org.springframework.context.annotation.Scope;
733     import org.springframework.stereotype.Component;
734
735     @Component
736     public class ApplicationConfig {
737         @Bean
738         @Scope("prototype")
739         public Student student() {
740             Student student = new Student("한라산", 35);
741             return student;
742         }
743     }
744

```

```

745     4)com.example > right-click > New > Class
746     5)Name : MainClass2
747     6)Finish
748

```

```

749     package com.example;
750
751     import
752     org.springframework.context.annotation.AnnotationConfigApplicationContext;
753
754     public class MainClass2 {
755         public static void main(String[] args) {
756             AnnotationConfigApplicationContext context =
757                 new AnnotationConfigApplicationContext(ApplicationConfig.class);
758
759             Student student = context.getBean("student", Student.class);
760             System.out.println(student);
761             System.out.println("-----");
762
763             Student student1 = context.getBean("student", Student.class);

```

```
763         student1.setName("북한산");
764         student1.setAge(55);
765         System.out.println(student1);
766         System.out.println("-----");
767
768         if(student.equals(student1)) System.out.println("Equals"); //Print Equals
769         else System.out.println("Different");
770         context.close();
771     }
772 }
```

#### 7)실행결과

```
775
776     Student(name=한라산, age=35)
777     -----
778     Student(name=북한산, age=55)
779     -----
780     Different
```