

```
1  HOL : Spring DI
2  -----
3  Task 1. Non-DI Java Project
4  1. Project 유형 : Java Project
5  2. Project Name : BeforeSpring
6  3. Package Name : com.example
7
8
9  4. Calculator Class
10 com.example.Calculator.java
11 package com.example;
12
13 public class Calculator {
14     public void addAction(int a, int b){
15         System.out.println("Called addAction()");
16         System.out.printf("%d + %d = %d\n", a, b, (a + b));
17     }
18     public void subAction(int a, int b){
19         System.out.println("Called subAction()");
20         System.out.printf("%d - %d = %d\n", a, b, (a - b));
21     }
22     public void multiAction(int a, int b){
23         System.out.println("Called multiAction()");
24         System.out.printf("%d x %d = %d\n", a, b, (a * b));
25     }
26     public void divAction(int a, int b){
27         System.out.println("Called divAction()");
28         System.out.printf("%d / %d = %d\n", a, b, (a / b));
29     }
30 }
31
32
33 5. MyCalculator Class
34 com.example.MyCalculator.java
35 package com.example;
36
37 public class MyCalculator {
38     private Calculator calculator;
39     private int firstNum;
40     private int secondNum;
41
42     public void setFirstNum(int firstNum) {
43         this.firstNum = firstNum;
44     }
45     public void setSecondNum(int secondNum) {
46         this.secondNum = secondNum;
47     }
48     public void setCalculator(Calculator calculator){
49         this.calculator = calculator;
50     }
51
52     public void add(){
53         this.calculator.addAction(firstNum, secondNum);
54     }
55     public void sub(){
56         this.calculator.subAction(firstNum, secondNum);
57     }
58     public void multi(){
59         this.calculator.multiAction(firstNum, secondNum);
60     }
61     public void div(){
62         this.calculator.divAction(firstNum, secondNum);
63     }
64 }
65
66
67 6. MainClass Class
```

```

68 com.example.MainClass
69 package com.example;
70
71 public class MainClass {
72     public static void main(String[] args) {
73         MyCalculator myCalculator = new MyCalculator();
74         myCalculator.setCalculator(new Calculator());
75
76         myCalculator.setFirstNum(10);
77         myCalculator.setSecondNum(2);
78
79         myCalculator.add();
80         myCalculator.sub();
81         myCalculator.multi();
82         myCalculator.div();
83     }
84 }
85
86

```

87 7. Result

```

88 Called addAction()
89 10 + 2 = 12
90 Called subAction()
91 10 - 2 = 8
92 Called multiAction()
93 10 x 2 = 20
94 Called divAction()
95 10 / 2 = 5
96
97
98
99 -----

```

100 Task 2. DI Demo in Spring

101 1. New > Java Project

```

102 1)Project Name : StartSpring
103 2)JRE : Use default JRE 'jdk-11.0.12' and workspace compiler preferences
104 3)Uncheck [Create module-info.java file]
105 4)Next
106 5)Finish
107

```

108 2. Create package to src : com.example

109 3. Copy MyCalculator.java, Calculator.java from BeforeSpring project to StartSpring's package

112 4. Create class : com.example.MainClass.java

```

113 package com.example;
114
115 public class MainClass {
116     public static void main(String[] args) {
117
118     }
119 }
120
121

```

122 5. Java Project를 Spring Project로 변환

123 1)StartSpring Project > right-click > Configure > Convert to Maven Project

```

124 -Project : /StartSpring
125 -Group Id : StartSpring
126 -Artifact Id : StartSpring
127 -version : 0.0.1-SNAPSHOT
128 -Packaging : jar
129 -Finish
130 -Package Explorer에서 보이는 Project icon에 Maven의 'M'자가 보임.
131

```

132 2)StartSpring Project > right-click > Spring > Add Spring Project Nature

```

133 -Package Explorer에서 보이는 Project icon에 'M'자와 Spring의 'S'가 보임.
134

```

3) pom.xml file에 Spring Context Dependency 추가하기
- <https://mvnrepository.com>에서 'spring context'로 검색
-[Spring Context] click
- 현재 Spring 5.x의 현재 version인 5.3.10 click
- Copy하여 pom.xml에 paste

```
<version>0.0.1-SNAPSHOT</version>
<dependencies> <--- dependencies element 추가
  <dependency> <--- 여기에 paste
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.10</version>
  </dependency>
</dependencies>
```

4) pom.xml Save

5) pom.xml > right-click > Run As > Maven install
[INFO] BUILD SUCCESS 확인

6. config folder 생성

1) StartSpring project > right-click > New > Source Folder
- Folder name : config
- Finish

7. Bean Configuration XML 작성

1) config > right-click > New > Other > Spring > Spring Bean Configuration File > Next
2) Name : applicationContext.xml > 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"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

  <bean id="calculator" class="com.example.Calculator" />

  <bean id="myCalculator" class="com.example.MyCalculator">
    <property name="calculator">
      <ref bean="calculator" />
    </property>
    <property name="firstNum" value="10" />
    <property name="secondNum" value="2" />
  </bean>
</beans>
```

8. MainClass.java

```
package com.example;

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

public class MainClass {
  public static void main(String[] args) {
    String configFile = "classpath:applicationContext.xml";
    AbstractApplicationContext ctx = new GenericXmlApplicationContext(configFile);
    MyCalculator myCalculator = ctx.getBean("myCalculator", MyCalculator.class);

    myCalculator.add();
    myCalculator.sub();
    myCalculator.multi();
    myCalculator.div();

    ctx.close();
  }
}
```

201 }

202

203

204 9. Result

205 BeforeSpring과 같음.

206 Called addAction()

207 $10 + 2 = 12$

208 Called subAction()

209 $10 - 2 = 8$

210 Called multiAction()

211 $10 \times 2 = 20$

212 Called divAction()

213 $10 / 2 = 5$

214

215

216

217 -----

218 Task 3. 간단한 DI Project

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

220 1)Project name : DIDemo

221 2)JRE : Use default JRE 'jdk-11.0.12' and workspace compiler preferences

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

223 4)Next

224 5)Finish

225

226

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

228 1)Package name : com.example

229 2)Finish

230

231

232 3. Interface 작성

233 1)com.example > right-click > New > Interface

234 2)Name : Printer

235

236 3)Printer.java

237 package com.example;

238

239 public interface Printer{

240 void print(String message);

241 }

242

243

244 4. POJO class 작성

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

246 2)Name : Hello

247 3)Finish

248 4)Hello.java

249 package com.example;

250

251 public class Hello{

252 private String name;

253 private Printer printer;

254

255 public Hello(){}

256

257 public void setName(String name){

258 this.name = name;

259 }

260

261 public void setPrinter(Printer printer){

262 this.printer = printer;

263 }

264

265 public String sayHello(){

266 return "Hello " + name;

267 }

```

268
269     public void print(){
270         this.printer.print(sayHello());
271     }
272 }
273
274

```

5. Printer interface의 child class 작성하기

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

```

276     -Name : StringPrinter
277     -Interfaces : com.example.Printer
278     -Finish
279
280

```

2) StringPrinter.java

```

281     package com.example;
282
283
284     public class StringPrinter implements Printer{
285         private StringBuffer buffer = new StringBuffer();
286
287         @Override
288         public void print(String message){
289             this.buffer.append(message);
290         }
291
292         public String toString(){
293             return this.buffer.toString();
294         }
295     }
296

```

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

```

297     -Name : ConsolePrinter
298     -Interface : com.example.Printer
299     -Finish
300
301

```

4) ConsolePrinter.java

```

302     package com.example;
303
304
305     public class ConsolePrinter implements Printer{
306
307         @Override
308         public void print(String message){
309             System.out.println(message);
310         }
311     }
312
313

```

6. Java Project를 Spring Project로 변환

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

```

314     -Project : /DIDemo
315     -Group Id : DIDemo
316     -Artifact Id : DIDemo
317     -version : 0.0.1-SNAPSHOT
318     -Packaging : jar
319     -Finish
320
321

```

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

3) pom.xml file에 Spring Context Dependency 추가하기

```

322
323
324
325     <version>0.0.1-SNAPSHOT</version>
326     <dependencies>
327         <dependency>
328             <groupId>org.springframework</groupId>
329             <artifactId>spring-context</artifactId>
330             <version>5.3.10</version>
331         </dependency>
332     </dependencies>
333
334

```

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

[INFO] BUILD SUCCESS 확인

7. config folder 생성

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

-Folder name : config

-Finish

8. Bean Configuration XML 작성

1)config > right-click > New > Other > Spring > Spring Bean Configuration File > Next

2)File name : beans.xml

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"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

  <bean id="hello" class="com.example.Hello">
    <property name="name" value="Spring" />
    <property name="printer" ref="printer" />
  </bean>
  <bean id="printer" class="com.example.StringPrinter" />
  <bean id="consolePrinter" class="com.example.ConsolePrinter" />

</beans>
```

9. Beans Graph 사용하기

1)Window menu > Show View > Other > Spring > Spring Explorer > Open

2)Spring Explorer

-DIDemo > Beans > beans.xml > right-click > Open Beans Graph

10. DI Test class 작성

1)src/com.example > right-click > New > Package

-Name : com.example.test

-Finish

2)/src/com.example.test > New > Class

-Name : HelloBeanTest.java

```
package com.example.test;

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

import com.example.Hello;
import com.example.Printer;

public class HelloBeanTest {
  public static void main(String [] args){
    //1. IoC Container 생성
    ApplicationContext context =
      new GenericXmlApplicationContext("classpath:beans.xml");

    //2. Hello Beans 가져오기
    Hello hello = (Hello)context.getBean("hello");
    System.out.println(hello.sayHello());
    hello.print();

    //3. SpringPrinter 가져오기
    Printer printer = (Printer)context.getBean("printer");
    System.out.println(printer.toString());
```

```

401
402         Hello hello2 = context.getBean("hello", Hello.class);
403         hello2.print();
404
405         System.out.println(hello == hello2); //Singleton Pattern
406     }
407 }
408
409

```

410 11. Result

```

411     Hello Spring
412     Hello Spring
413     true
414
415
416

```

417 -----

418 Task 4. JUnit을 사용한 DI test class 작성하기

419 1. JUnit을 사용한 DI test class(HelloBeanJUnitTest.java) 작성

420 1) pom.xml에 아래 코드 붙여넣기

```

421     <dependency>
422         <groupId>junit</groupId>
423         <artifactId>junit</artifactId>
424         <version>4.13.2</version>
425         <scope>test</scope>
426     </dependency>
427

```

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

429 [INFO] BUILD SUCCESS 확인

431 3) src/com.example.test > New > Class

432 -Name : HelloBeanJUnitTest.java

```

433
434     package com.example.test;
435
436     import org.junit.Before;
437     import org.junit.Test;
438     import org.springframework.context.ApplicationContext;
439     import org.springframework.context.support.GenericXmlApplicationContext;
440
441     import com.example.Hello;
442     import com.example.Printer;
443
444     import static org.junit.Assert.assertEquals;
445     import static org.junit.Assert.assertSame;
446
447     public class HelloBeanJUnitTest {
448         private ApplicationContext context;
449
450         @Before
451         public void init(){
452             //항상 먼저 ApplicationContext를 생성해야 하기 때문에
453             //1. IoC Container 생성
454             context = new GenericXmlApplicationContext("classpath:beans.xml");
455         }
456
457         @Test
458         public void test1(){
459             //2. Hello Beans 가져오기
460             Hello hello = (Hello)context.getBean("hello");
461             assertEquals("Hello Spring", hello.sayHello());
462             hello.print();
463
464             //3. SpringPrinter 가져오기
465             Printer printer = (Printer)context.getBean("printer");
466             assertEquals("Hello Spring", printer.toString());
467         }

```

```

468
469     @Test
470     public void test2(){
471         Hello hello = (Hello)context.getBean("hello");
472
473         Hello hello2 = context.getBean("hello", Hello.class);
474         assertEquals(hello, hello2);
475     }
476 }
477
478

```

479 2. @Before에 mouse를 올려놓으면 Fix project setup... click

480 1)Add archive 'junit-4.12.jar ... > OK

481 -import org.junit...에 mouse를 올려놓으면 Fix project setup... click

482 -Add JUnit 4 library to the build path > OK

483

484

485 3. right-click > Run As > JUnit Test

486 1)결과 -> JUnit View에 초록색 bar

487 2)만일, test1() method를 junit에서 제외하고 싶을 때에는 @Test 옆에 @Ignore를 선언한다.

488

```

489     import org.junit.Ignore;
490     ...
491     @Test @Ignore
492     public void test1(){
493     ...
494

```

490

491 @Test @Ignore

492 public void test1(){

493 ...

494

495 3)right-click > Run As > JUnit Test

496 -JUnit Test 목록에서 test1()는 실행되지 않는다.

497

498

499

500 -----

501 Task 5. Spring TestContext Framework

502 1. Spring-Test library 설치

503 1)http://mvnrepository.com에서 'spring test'로 검색

504 2)검색 결과 목록에서 'Spring TestContext Framework' Click

505 3)version 목록에서 5.3.10 Click

506

507

508 2. dependency 복사해서 pom.xml에 붙여넣기

```

509 <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
510 <dependency>
511     <groupId>org.springframework</groupId>
512     <artifactId>spring-test</artifactId>
513     <version>5.3.10</version>
514     <scope>test</scope>
515 </dependency>
516
517

```

510 <dependency>

511 <groupId>org.springframework</groupId>

512 <artifactId>spring-test</artifactId>

513 <version>5.3.10</version>

514 <scope>test</scope>

515 </dependency>

516

517

518 3. pom.xml > right-click > Maven Install

519 [INFO] BUILD SUCCESS

520

521

522 4. Spring-Test를 사용할 DI test class-HelloBeanJUnitSpringTest.java 작성하기

523 1)src/com.example.test > New > Class

524 2)Name : HelloBeanJUnitSpringTest

```

525     package com.example.test;
526
527     import static org.junit.Assert.assertEquals;
528     import static org.junit.Assert.assertEquals;
529
530     import org.junit.Test;
531     import org.junit.runner.RunWith;
532     import org.springframework.beans.factory.annotation.Autowired;
533     import org.springframework.context.ApplicationContext;
534     import org.springframework.test.context.ContextConfiguration;

```

526

527 import static org.junit.Assert.assertEquals;

528 import static org.junit.Assert.assertEquals;

529

530 import org.junit.Test;

531 import org.junit.runner.RunWith;

532 import org.springframework.beans.factory.annotation.Autowired;

533 import org.springframework.context.ApplicationContext;

534 import org.springframework.test.context.ContextConfiguration;


```

import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

import com.example.Hello;
import com.example.Printer;

@RunWith(SpringJUnit4ClassRunner.class)
//JUnit 4.x에서 사용
@ContextConfiguration(locations="classpath:beans.xml")
public class HelloBeanJUnitSpringTest {
    @Autowired
    ApplicationContext context;

    @Test
    public void test1(){
        Hello hello = (Hello)context.getBean("hello");
        assertEquals("Hello Spring", hello.sayHello());
        hello.print();

        Printer printer = (Printer)context.getBean("printer");
        assertEquals("Hello Spring", printer.toString());
    }

    @Test
    public void test2(){
        Hello hello = (Hello)context.getBean("hello");

        Hello hello2 = context.getBean("hello", Hello.class);
        assertEquals(hello, hello2);
    }
}

```

-right-click > Run As > JUnit Test
 -결과 -> JUnit View에 초록색 bar

- 4)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
 -해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
 -spring-test-5.3.10.jar 선택 후 [Remove] 로 삭제
 -Classpath 선택
 -[Add External JARs...] Click
 -Local M2 Repository(e.g
 C:\Users\instructor\.m2\repository\org\springframework\spring-test\5.3.10\spring-test-5.3.10.jar
)에서 직접 jar(spring-test-5.3.10.jar)를 선택할 것
 -[Order and Export] tab에서 spring-test-5.3.10.jar 선택 후 [Up] button을 클릭
 -해당 DIDemo/src 바로 아래까지 올리고 [Apply and Close] Click

 Task 6. Java Annotation을 이용하여 setter를 이용한 의존주입하기 실습

1. In Package Explorer > right-click > New > Java Project
 - 1)Project name : DIDemo1
 - 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
 - 2)Finish
3. POJO class 작성
 - 1)com.example > right-click > New > Class
 - 2)Name : Hello

```

600 package com.example;
601
602 public class Hello {
603     private String name;
604     private Printer printer;
605
606     public Hello(){ }
607
608     public void setName(String name){
609         this.name = name;
610     }
611
612     public void setPrinter(Printer printer){
613         this.printer = printer;
614     }
615
616     public String sayHello(){
617         return "Hello " + name;
618     }
619
620     public void print(){
621         this.printer.print(sayHello());
622     }
623 }
624

```

625 3)com.example > right-click > New > Interface

626 4)Name : Printer

```

627
628 package com.example;
629
630 public interface Printer{
631     void print(String message);
632 }
633

```

634 5)com.example > right-click > New > Class

635 6)Name : StringPrinter

636 7)Interfaces : com.example.Printer

```

637
638 package com.example;
639
640 public class StringPrinter implements Printer{
641     private StringBuffer buffer = new StringBuffer();
642
643     @Override
644     public void print(String message){
645         this.buffer.append(message);
646     }
647
648     public String toString(){
649         return this.buffer.toString();
650     }
651 }
652

```

653 8)com.example > right-click > New > Class

654 9)Name : ConsolePrinter

655 10)Interfaces : com.example.Printer

```

656
657 package com.example;
658
659 public class ConsolePrinter implements Printer{
660
661     @Override
662     public void print(String message){
663         System.out.println(message);
664     }
665 }
666

```

```
667
668 4. Java Project를 Spring Project로 변환
669   1)DIDemo1 Project > right-click > Configure > Convert to Maven Project
670     -Project : /DIDemo1
671     -Group Id : DIDemo1
672     -Artifact Id : DIDemo1
673     -version : 0.0.1-SNAPSHOT
674     -Packaging : jar
675     -Finish
676
677   2)DIDemo1 Project > right-click > Spring > Add Spring Project Nature
678
679   3)pom.xml file에 Spring Context Dependency 추가하기
680     <version>0.0.1-SNAPSHOT</version>
681     <dependencies>
682       <dependency>
683         <groupId>org.springframework</groupId>
684         <artifactId>spring-context</artifactId>
685         <version>5.3.10</version>
686       </dependency>
687     </dependencies>
688
689   4)pom.xml > right-click > Run As > Maven install
690     [INFO] BUILD SUCCESS
691
692
693 5. config package 생성
694   1)com.example > right-click > New > Package > com.example.config
695   2)Finish
696
697
698 6. AppCtx Class 생성
699   1)com.example.config > right-click > New > Class
700   2)Name : AppCtx.java
701
702     package com.example.config;
703
704     import org.springframework.context.annotation.Bean;
705     import org.springframework.context.annotation.Configuration;
706
707     import com.example.ConsolePrinter;
708     import com.example.Hello;
709     import com.example.StringPrinter;
710
711     @Configuration
712     public class AppCtx {
713
714         @Bean
715         public Hello hello() {
716             Hello hello = new Hello();
717             hello.setName("Spring");
718             hello.setPrinter(this.printer());
719             return hello;
720         }
721
722         @Bean
723         public StringPrinter printer() {
724             return new StringPrinter();
725         }
726
727         @Bean
728         public ConsolePrinter consolePrinter() {
729             return new ConsolePrinter();
730         }
731     }
732
733
```

```

734 7. DI Test class 작성
735 1)src > right-click > New > Package
736 2)Package Name : com.example.test
737 3)Finish
738 4)com.example.test > right-click > New > Class
739 5)Name : HelloBeanTest
740
741 package com.example.test;
742
743 import org.springframework.context.ApplicationContext;
744 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
745
746 import com.example.Hello;
747 import com.example.Printer;
748 import com.example.config.AppCtx;
749
750 public class HelloBeanTest {
751     public static void main(String[] args) {
752         // 1. IoC Container 생성
753         ApplicationContext ctx = new AnnotationConfigApplicationContext(AppCtx.class);
754
755         // 2. Hello Beans 가져오기
756         Hello hello = (Hello)ctx.getBean("hello");
757         System.out.println(hello.sayHello());
758         hello.print();
759
760         // 3. SpringPrinter 가져오기
761         Printer printer = (Printer) ctx.getBean("printer");
762         System.out.println(printer.toString());
763         Hello hello2 = ctx.getBean("hello", Hello.class);
764         hello2.print();
765         System.out.println(hello == hello2);
766     }
767 }

```

770 8. Test

```

771 1)/src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application
772 Hello Spring
773 Hello Spring
774 true
775
776
777

```

778 -----

779 Task 7. setter를 이용한 의존주입하기 실습

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

```

781 1)Project Name : SpringDemo
782 2)JRE
783 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
784 3)Uncheck [Create module-info.java file]
785 4)Next
786 5)Finish
787
788

```

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

```

790 1)Package name : com.example
791
792

```

793 3. POJO class 작성

```

794 1)com.example > right-click > New > Class
795 2)Class Name : BmiCalculator
796

```

```

796 package com.example;
797
798
799 public class BmiCalculator {
800     private double lowWeight;

```

```

801     private double normal;
802     private double overWeight;
803     private double obesity;
804
805     public void setLowWeight(double lowWeight) {
806         this.lowWeight = lowWeight;
807     }
808
809     public void setNormal(double normal) {
810         this.normal = normal;
811     }
812
813     public void setOverWeight(double overWeight) {
814         this.overWeight = overWeight;
815     }
816
817     public void setObesity(double obesity) {
818         this.obesity = obesity;
819     }
820     public void bmiCalcu(double weight, double height){
821         double h = height * 0.01;
822         double result = weight / (h * h);
823
824         System.out.println("BMI 지수 : " + (int)result);
825
826         if(result > obesity)
827             System.out.println("비만입니다.");
828         else if(result > overWeight)
829             System.out.println("과체중입니다.");
830         else if(result > normal)
831             System.out.println("정상입니다.");
832         else
833             System.out.println("저체중입니다.");
834     }
835 }

```

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

4)Class Name : MyInfo.java

```

840     package com.example;
841
842     import java.util.ArrayList;
843
844     public class MyInfo {
845         private String name;
846         private double height;
847         private double weight;
848         private ArrayList<String> hobby;
849         private BmiCalculator bmiCalculator;
850
851         public void setBmiCalculator(BmiCalculator bmiCalculator) {
852             this.bmiCalculator = bmiCalculator;
853         }
854         public void setName(String name) {
855             this.name = name;
856         }
857         public void setHeight(double height) {
858             this.height = height;
859         }
860         public void setWeight(double weight) {
861             this.weight = weight;
862         }
863         public void setHobby(ArrayList<String> hobby) {
864             this.hobby = hobby;
865         }
866         public void getInfo(){
867             System.out.println("Name : " + this.name);

```

```

868         System.out.println("Height : " + this.height);
869         System.out.println("Weight : " + this.weight);
870         System.out.println("Hobby : " + this.hobby);
871         this.bmiCalcu();
872     }
873     public void bmiCalcu(){
874         this.bmiCalculator.bmiCalcu(this.weight, this.height);
875     }
876 }
877
878

```

879 4. Java Project를 Spring Project로 변환

880 1)SpringDemo Project > right-click > Configue > Convert to Maven Project

```

881     -Project : /SpringDemo
882     -Group Id : SpringDemo
883     -Artifact Id : SpringDemo
884     -version : 0.0.1-SNAPSHOT
885     -Packaging : jar
886     -Finish
887

```

888 2)SpringDemo Project > right-click > Spring > Add Spring Project Nature

889

890 3)pom.xml file에 Spring Context Dependency 추가하기

```

891     <version>0.0.1-SNAPSHOT</version>
892     <dependencies>
893         <dependency>
894             <groupId>org.springframework</groupId>
895             <artifactId>spring-context</artifactId>
896             <version>5.3.10</version>
897         </dependency>
898     </dependencies>
899

```

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

901 [INFO] BUILD SUCCESS 확인

902

903

904 5. SpringDemo/resources folder 생성

905 1)SpringDemo project > right-click > Build Path > Configure Build Path

906 2)Source Tab > Add Folder

907 3)SpringDemo 선택 확인

908 4)Create New Folder > Folder name : resources > Finish > OK

909 5)SpringDemo/resources(new) 확인

910 6)Apply and Close

911

912

913 6. Bean Configuration XML 작성

914 1)SpringDemo/resources > right-click > New > Other > Spring > Spring Bean Configuration File

915 2)File name : applicationContext.xml

916 3)Finish

917

```

918     <bean id="bmiCalculator" class="com.example.BmiCalculator">
919         <property name="lowWeight" value="18.5" />
920         <property name="normal" value="23" />
921         <property name="overWeight" value="25" />
922         <property name="obesity">
923             <value>30</value>
924         </property>
925     </bean>
926     <bean id="myInfo" class="com.example.MyInfo">
927         <property name="name" value="백두산" />
928         <property name="height" value="170.5" />
929         <property name="weight" value="67" />
930         <property name="hobby">
931             <list>
932                 <value>수영</value>
933                 <value>요리</value>
934                 <value>독서</value>

```

```

935         </list>
936     </property>
937     <property name="bmiCalculator">
938         <ref bean="bmiCalculator" />
939     </property>
940 </bean>

```

7. MainClass 생성하기

```

944 1)com.example.MainClass.java
945     package com.example;

```

```

947     import org.springframework.context.AbstractApplicationContext;
948     import org.springframework.context.support.GenericXmlApplicationContext;
949
950     public class MainClass {
951         public static void main(String[] args) {
952             String configFile = "classpath:applicationContext.xml";
953
954             //Spring Container 생성
955             AbstractApplicationContext context = new GenericXmlApplicationContext(configFile);
956
957             //Spring Container 에서 객체를 가져옴
958             MyInfo myInfo = context.getBean("myInfo", MyInfo.class);
959
960             myInfo.getInfo();
961             context.close();
962         }
963     }

```

8. Java Application 실행

```

967     Name : 백두산
968     Height : 170.5
969     Weight : 67.0
970     Hobby : [수영, 요리, 독서]
971     BMI 지수 : 23
972     정상입니다.

```

[추가 lab] : PropertyEditor 실습

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

```

978     1)Project Name : PropertyEditorDemo
979     2)JRE
980         -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
981     3)Uncheck [Create module-info.java file]
982     4)Next
983     5)Finish

```

2. src > right-click > New > Package

```

987     1)Package name : com.example

```

3. POJO class 작성

```

991     1)com.example > right-click > New > Class
992     2)Class Name : SimpleBean

```

```

994     package com.example;
995
996     import java.io.File;
997     import java.io.InputStream;
998     import java.net.URL;
999     import java.util.Date;
1000     import java.util.List;
1001     import java.util.Locale;

```

```

1002 import java.util.Properties;
1003 import java.util.regex.Pattern;
1004
1005 public class SimpleBean {
1006     private byte[] bytes; // ByteArrayPropertyEditor
1007     private Class cls; // ClassEditor
1008     private Boolean trueOrFalse; // CustomBooleanEditor
1009     private List<String> stringList; // CustomCollectionEditor
1010     private Float floatValue; // CustomNumberEditor
1011     private File file; // CustomFileEditor
1012     private InputStream stream; // InputStreamEditor
1013     private Locale locale; // LocaleEditor
1014     private Pattern pattern; // PatternEditor
1015     private Properties properties; // PropertiesEditor
1016     private URL url; // URLEditor
1017
1018     public void setBytes(byte[] bytes) {
1019         System.out.println("Adding " + bytes.length + "bytes");
1020         this.bytes = bytes;
1021     }
1022
1023     public void setCls(Class cls) {
1024         System.out.println("Setting class : " + cls.getName());
1025         this.cls = cls;
1026     }
1027
1028     public void setTrueOrFalse(Boolean trueOrFalse) {
1029         System.out.println("Settting Boolean : " + trueOrFalse);
1030         this.trueOrFalse = trueOrFalse;
1031     }
1032
1033     public void setStringList(List<String> stringList) {
1034         System.out.println("Setting string list with size : " + stringList.size());
1035         for (String s : stringList) {
1036             System.out.println("String member : " + s);
1037         }
1038         this.stringList = stringList;
1039     }
1040
1041     public void setFloatValue(Float floatValue) {
1042         System.out.println("Setting float value : " + floatValue);
1043         this.floatValue = floatValue;
1044     }
1045
1046     public void setFile(File file) {
1047         System.out.println("Setting file : " + file.getName());
1048         this.file = file;
1049     }
1050
1051     public void setStream(InputStream stream) {
1052         System.out.println("Setting stream : " + stream);
1053         this.stream = stream;
1054     }
1055
1056     public void setLocale(Locale locale) {
1057         System.out.println("Setting Locale : " + locale.getDisplayName());
1058         this.locale = locale;
1059     }
1060
1061     public void setPattern(Pattern pattern) {
1062         System.out.println("Setting pattern : " + pattern);
1063         this.pattern = pattern;
1064     }
1065
1066     public void setProperties(Properties properties) {
1067         System.out.println("Loaded : " + properties.size() + "properties");
1068         this.properties = properties;

```



```

1069     }
1070
1071     public void setUrl(URL url) {
1072         System.out.println("Setting URL : " + url.toExternalForm());
1073         this.url = url;
1074     }
1075
1076 }
1077
1078

```

4. Java Project를 Spring Project로 변환

1)PropertyEditorDemo Project > right-click > Configue > Convert to Maven Project

```

-Project : /PropertyEditorDemo
-Group Id : PropertyEditorDemo
-Artifact Id : PropertyEditorDemo
-version : 0.0.1-SNAPSHOT
-Packaging : jar
-Finish

```

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

3)pom.xml file에 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. PropertyEditorDemo/resources folder 생성

1)PropertyEditorDemo project > right-click > Build Path > Configure Build Path

2)Source Tab > Add Folder

3)PropertyEditorDemo 선택 확인

4>Create New Folder > Folder name : resources > Finish > OK

5)PropertyEditorDemo/resources(new) 확인

6)Apply and Close

6. Bean Configuration XML 작성

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

2)File name : applicationContext.xml

3)Finish

```

<!-- 실제 bean에 대한 정의 -->
<bean id="simpleBean" class="com.example.SimpleBean">
  <!-- property type에 맞게 알아서 PropertyEditor가 동작한다. -->
  <property name="bytes">
    <value>Hello, World</value>
  </property>
  <property name="cls">
    <value>java.lang.String</value>
  </property>
  <property name="trueOrFalse">
    <value>true</value>
  </property>
  <property name="stringList">
    <util:list>
      <value>String member 1</value>
      <value>String member 2</value>
    </util:list>
  </property>
</bean>

```

```

1135     </property>
1136     <property name="floatValue">
1137         <value>123.45678</value>
1138     </property>
1139     <property name="file">
1140         <value>classpath:applicationContext.xml</value>
1141     </property>
1142     <property name="stream">
1143         <value>classpath:applicationContext.xml</value>
1144     </property>
1145     <property name="locale">
1146         <value>en_US</value>
1147     </property>
1148     <property name="pattern">
1149         <value>a*b</value>
1150     </property>
1151     <property name="properties">
1152         <value>
1153             name=foo
1154             age=19
1155         </value>
1156     </property>
1157     <property name="url">
1158         <value>http://java.sun.com</value>
1159     </property>
1160 </bean>
1161
1162

```

7. MainClass 생성하기

1) com.example.MainClass.java

```

1165     import org.springframework.context.support.GenericXmlApplicationContext;
1166
1167     public class MainClass {
1168         public static void main(String[] args) {
1169             GenericXmlApplicationContext ctx = new GenericXmlApplicationContext();
1170             ctx.load("classpath:applicationContext.xml");
1171             ctx.registerShutdownHook();
1172             ctx.refresh();
1173         }
1174     }
1175
1176

```

8. Java Application 실행

Task 8. 생성자 이용하여 의존 주입하기 실습

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

1) Project name : DIDemo2

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

2) Finish

3. POJO class 작성

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

2) Class Name : Hello

package com.example;

```

1200     public class Hello{
1201         private String name;

```

```

1202     private Printer printer;
1203
1204     public Hello(){ }
1205
1206     public void setName(String name){
1207         this.name = name;
1208     }
1209
1210     public void setPrinter(Printer printer){
1211         this.printer = printer;
1212     }
1213
1214     public String sayHello(){
1215         return "Hello " + name;
1216     }
1217
1218     public void print(){
1219         this.printer.print(sayHello());
1220     }
1221 }
1222

```

1223 3)com.example > right-click > New > Interface

1224 4)interface name : Printer

```

1225
1226     package com.example;
1227
1228     public interface Printer{
1229         void print(String message);
1230     }
1231

```

1232 5)com.example > right-click > New > Class

1233 6)Class Name : StringPrinter

1234 7)Interfaces : com.example.Printer

```

1235
1236     package com.example;
1237
1238     public class StringPrinter implements Printer{
1239         private StringBuffer buffer = new StringBuffer();
1240
1241         @Override
1242         public void print(String message){
1243             this.buffer.append(message);
1244         }
1245
1246         public String toString(){
1247             return this.buffer.toString();
1248         }
1249     }
1250

```

1251 8)com.example > right-click > New > Class

1252 9)Class Name : ConsolePrinter

1253 10)Intefaces : com.example.Printer

```

1254
1255     package com.example;
1256
1257     public class ConsolePrinter implements Printer{
1258
1259         @Override
1260         public void print(String message){
1261             System.out.println(message);
1262         }
1263     }
1264
1265

```

1266 4. Java Project를 Spring Project로 변환

1267 1)DIDemo2 Project > right-click > Configure > Convert to Maven Project

1268 -Project : /DIDemo2

```
1269 -Group Id : DIDemo2
1270 -Artifact Id : DIDemo2
1271 -version : 0.0.1-SNAPSHOT
1272 -Packaging : jar
1273 -Finish
1274
1275 2)DIDemo2 Project > right-click > Spring > Add Spring Project Nature
1276
1277 3)pom.xml file에 Spring Context Dependency 추가하기
1278 <version>0.0.1-SNAPSHOT</version>
1279 <dependencies>
1280 <dependency>
1281 <groupId>org.springframework</groupId>
1282 <artifactId>spring-context</artifactId>
1283 <version>5.3.10</version>
1284 </dependency>
1285 </dependencies>
1286
1287 4)pom.xml > right-click > Run As > Maven install
1288 [INFO] BUILD SUCCESS 확인
1289
1290
1291 5. DIDemo2/resources folder 생성
1292 1)DIDemo2 project > right-click > Build Path > Configure Build Path
1293 2)Source Tab > Add Folder
1294 3)DIDemo2 선택확인
1295 4)Create New Folder > Folder name : resources > Finish > OK
1296 5)DIDemo2/resources(new) 확인
1297 6)Apply and Close
1298
1299
1300 6. Bean Configuration XML 작성
1301 1)DIDemo2/resources > right-click > New > Other > Spring > Spring Bean Configuration File
1302 -File name : beans.xml > Finish
1303
1304 <?xml version="1.0" encoding="UTF-8"?>
1305 <beans xmlns="http://www.springframework.org/schema/beans"
1306 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1307 xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
1308
1309 <bean id="hello" class="com.example.Hello">
1310 <property name="name" value="Spring" />
1311 <property name="printer" ref="printer" />
1312 </bean>
1313 <bean id="printer" class="com.example.StringPrinter" />
1314 <bean id="consolePrinter" class="com.example.ConsolePrinter" />
1315
1316 </beans>
1317
1318
1319 7. Test class 작성
1320 1)/src > right-click > New > Package
1321 2)Package Name : com.example.test
1322 3)/src/com.example/test/HelloBeanTest.java
1323
1324 package com.example.test;
1325
1326 import org.springframework.context.ApplicationContext;
1327 import org.springframework.context.support.GenericXmlApplicationContext;
1328
1329 import com.example.Hello;
1330 import com.example.Printer;
1331
1332 public class HelloBeanTest {
1333     public static void main(String [] args){
1334         //1. IoC Container 생성
```

```

1335     ApplicationContext context =
1336         new GenericXmlApplicationContext("classpath:beans.xml");
1337
1338     //2. Hello Beans 가져오기
1339     Hello hello = (Hello)context.getBean("hello");
1340     System.out.println(hello.sayHello());
1341     hello.print();
1342
1343     //3. SpringPrinter 가져오기
1344     Printer printer = (Printer)context.getBean("printer");
1345     System.out.println(printer.toString());
1346
1347     Hello hello2 = context.getBean("hello", Hello.class);
1348     hello2.print();
1349
1350     System.out.println(hello == hello2); //Singleton Pattern
1351 }
1352 }
1353
1354
1355 8. Test
1356 1)/src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application
1357     Hello Spring
1358     Hello Spring
1359     true
1360
1361
1362 9. /src/com.example.Hello 생성자 추가
1363
1364     public Hello(String name, Printer printer) {
1365         this.name = name;
1366         this.printer = printer;
1367     }
1368
1369
1370 10. /resources/beans.xml에 아래 Code 추가
1371
1372     <bean id="hello2" class="com.example.Hello">
1373         <constructor-arg index="0" value="Spring" />
1374         <constructor-arg index="1" ref="printer" />
1375     </bean>
1376
1377
1378 11. /src/com.example.test/HelloBeanTest.java 수정
1379
1380     ...
1381     //2. Hello Beans 가져오기
1382     Hello hello = (Hello)context.getBean("hello2");
1383     ...
1384     Hello hello2 = context.getBean("hello2", Hello.class);
1385     ...
1386
1387
1388 12. Test
1389 1)/src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application
1390     Hello Spring
1391     Hello Spring
1392     true
1393
1394
1395 -----
1396
1397 Task 9. Java Annotation을 이용한 생성자 이용하여 의존 주입하기 실습
1398 1. In Package Explorer > right-click > New > Java Project
1399     1)Project Name : SpringDemo1
1400     2)JRE > Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
1401     3)Uncheck [Create module-info.java file]

```

1402 4)Next
1403 5)Finish
1404
1405
1406 2. src > right-click > New > Package
1407 1)Package name : com.example
1408 2)Finish
1409
1410
1411 3. POJO Class 생성
1412 1)com.example.Student.java
1413 package com.example;
1414
1415 public class Student {
1416 private String name;
1417 private int age;
1418 private int grade;
1419 private int classNum;
1420 }
1421
1422 2)com.example.StudentInfo.java
1423 package com.example;
1424
1425 public class StudentInfo {
1426 private Student student;
1427 }
1428
1429
1430 4. Java Project를 Spring Project로 변환
1431 1)SpringDemo1 Project > right-click > Configure > Convert to Maven Project
1432 -Project : /SpringDemo1
1433 -Group Id : SpringDemo1
1434 -Artifact Id : SpringDemo1
1435 -version : 0.0.1-SNAPSHOT
1436 -Packaging : jar
1437 -Finish
1438
1439 2)SpringDemo1 Project > right-click > Spring > Add Spring Project Nature
1440
1441 3)pom.xml file에 Spring Context Dependency 추가하기
1442 <version>0.0.1-SNAPSHOT</version>
1443 <dependencies>
1444 <dependency>
1445 <groupId>org.springframework</groupId>
1446 <artifactId>spring-context</artifactId>
1447 <version>5.3.10</version>
1448 </dependency>
1449 </dependencies>
1450
1451 4)pom.xml > right-click > Run As > Maven install
1452 [INFO] BUILD SUCCESS 확인
1453
1454
1455 5. Lombok library 추가
1456 1)<https://mvnrepository.com/>에서 'lombok'으로 검색
1457 2)'Project Lombok' click
1458 3)1.18.20 click
1459 4)dependency copy해서 pom.xml에 붙여넣기
1460
1461 <dependencies>
1462 <dependency>
1463 <groupId>org.springframework</groupId>
1464 <artifactId>spring-context</artifactId>
1465 <version>5.3.10</version>
1466 </dependency>
1467 <!-- <https://mvnrepository.com/artifact/org.projectlombok/lombok> -->
1468 <dependency>

```
1469         <groupId>org.projectlombok</groupId>
1470         <artifactId>lombok</artifactId>
1471         <version>1.18.20</version>
1472         <scope>provided</scope>
1473     </dependency>
1474 </dependencies>
1475
```

1476 5) pom.xml > right-click > Run As > Maven install
1477 [INFO] BUILD SUCCESS 확인
1478

1479 6. Student.java와 StudentInfo.java 수정

1480 1) Student.java

```
1481
1482     package com.example;
1483
1484     import lombok.Getter;
1485     import lombok.Setter;
1486     import lombok.ToString;
1487     import lombok.AllArgsConstructor;
1488
1489     @Getter
1490     @Setter
1491     @ToString
1492     @AllArgsConstructor
1493     public class Student {
1494         private String name;
1495         private int age;
1496         private int grade;
1497         private int classNum;
1498     }
1499
```

1500 2) StudentInfo.java

```
1501
1502     package com.example;
1503
1504     import lombok.Setter;
1505     import lombok.AllArgsConstructor;
1506
1507     @Setter
1508     @AllArgsConstructor
1509     public class StudentInfo {
1510         private Student student;
1511
1512         public void printInfo(){
1513             if(this.student != null){
1514                 System.out.println("Name : " + this.student.getName());
1515                 System.out.println("Age : " + this.student.getAge());
1516                 System.out.println("Grade : " + this.student.getGrade());
1517                 System.out.println("Class : " + this.student.getClassNum());
1518                 System.out.println("-----");
1519             }
1520         }
1521     }
1522
```

1523 7. 환경설정을 위해 config package 생성

- 1524 1) com.example package > right-click > New > Package
- 1525 2) Name : com.example.config
- 1526 3) Finish

1527 8. ApplicationContext.java 생성

- 1528 1) com.example.config > right-click > New > Class
- 1529 2) Name : ApplicationCtx
- 1530 3) Finish

```

1536 package com.example.config;
1537
1538 import org.springframework.context.annotation.Bean;
1539 import org.springframework.context.annotation.Configuration;
1540
1541 import com.example.Student;
1542 import com.example.StudentInfo;
1543
1544 @Configuration
1545 public class ApplicationCtx {
1546     @Bean
1547     public Student student1() {
1548         return new Student("백두산", 15, 2, 5);
1549     }
1550
1551     @Bean
1552     public Student student2() {
1553         return new Student("한라산", 16, 3, 7);
1554     }
1555
1556     @Bean
1557     public StudentInfo studentInfo() {
1558         return new StudentInfo(this.student1());
1559     }
1560 }

```

1561
1562
1563 9. com.example.MainClass.java

```

1564 package com.example;
1565
1566 import org.springframework.context.ApplicationContext;
1567 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
1568
1569 import com.example.config.ApplicationCtx;
1570
1571 public class MainClass {
1572     public static void main(String[] args) {
1573         ApplicationContext ctx = new AnnotationConfigApplicationContext(ApplicationCtx.class);
1574
1575         StudentInfo studentInfo = ctx.getBean("studentInfo", StudentInfo.class);
1576         studentInfo.printInfo();
1577
1578         Student student2 = ctx.getBean("student2", Student.class);
1579         studentInfo.setStudent(student2);
1580         studentInfo.printInfo();
1581     }
1582 }
1583
1584
1585

```

1586 10. Java Application 실행
1587 Name : 백두산
1588 Age : 15
1589 Grade : 2
1590 Class : 5
1591 -----
1592 Name : 한라산
1593 Age : 16
1594 Grade : 3
1595 Class : 7
1596 -----

1597
1598
1599
1600 -----

1601 Task 10. Context file 여러개 사용하기
1602 1. In Package Explorer > right-click > New > Java Project

1603 1)Project Name : SpringDemo2
1604 2)JRE
1605 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
1606 3)Uncheck [Create module-info.java file]
1607 4)Next
1608 5)Finish
1609
1610
1611 2)src > right-click > New > Package
1612 2)Package name : com.example
1613
1614
1615 3. POJO Class 생성
1616 1)com.example.Student.java
1617
1618 package com.example;
1619
1620 import java.util.ArrayList;
1621
1622 public class Student {
1623 private String name;
1624 private int age;
1625 private ArrayList<String> hobbies;
1626 private double height;
1627 private double weight;
1628 }
1629
1630 2)com.example.StudentInfo.java
1631
1632 package com.example;
1633 public class StudentInfo {
1634 private Student student;
1635 }
1636
1637 3)com.example.Product.java
1638
1639 package com.example;
1640 public class Product {
1641 private String pName;
1642 private int pPrice;
1643 private String maker;
1644 private String color;
1645 }
1646
1647
1648 4. Java Project를 Spring Project로 변환
1649 1)SpringDemo2 Project > right-click > Configure > Convert to Maven Project
1650 -Project : /SpringDemo2
1651 -Group Id : SpringDemo2
1652 -Artifact Id : SpringDemo2
1653 -version : 0.0.1-SNAPSHOT
1654 -Packaging : jar
1655 -Finish
1656
1657 2)SpringDemo2 Project > right-click > Spring > Add Spring Project Nature
1658
1659 3)pom.xml file에 Spring Context Dependency 추가하기
1660 <version>0.0.1-SNAPSHOT</version>
1661 <dependencies>
1662 <dependency>
1663 <groupId>org.springframework</groupId>
1664 <artifactId>spring-context</artifactId>
1665 <version>5.3.10</version>
1666 </dependency>
1667 </dependencies>
1668
1669 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. SpringDemo2/resources folder 생성

- 1) SpringDemo2 project > right-click > new > Source Folder
- 2) Folder Name : resources
- 3) Finish

7. Bean Configuration XML 작성

- 1) resources Folder > right-click > New > Spring Bean Configuration File
- 2) File name : applicationContext.xml > Finish
- 3) resources Folder > right-click > New > Spring Bean Configuration File
- 4) File name : applicationContext2.xml > Finish

8. Student.java, StudentInfo.java 그리고 Product.java에 lombok Annotation 붙이기

1) Student.java

```
package com.example;

import java.util.ArrayList;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NonNull;
import lombok.RequiredArgsConstructor;

@Data
@RequiredArgsConstructor
@AllArgsConstructor
public class Student {
    private @NonNull String name;
    private @NonNull int age;
    private @NonNull ArrayList<String> hobbies;
    private double height;
    private double weight;
}
```

2) StudentInfo.java

```
package com.example;
```

```

import lombok.Setter;
import lombok.Getter;

@Setter
@Getter
public class StudentInfo {
    private Student student;
}

```

3)Product.java

```

package com.example;

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

@NoArgsConstructor
@AllArgsConstructor
@RequiredArgsConstructor
@Setter
@ToString
public class Product {
    private @NonNull String pName;
    private @NonNull int pPrice;
    private String maker;
    private String color;
}

```

9. applicationContext.xml

```

<?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">

    <bean id="student1" class="com.example.Student">
        <constructor-arg value="백두산" />
        <constructor-arg value="25" />
        <constructor-arg>
            <list>
                <value>독서</value>
                <value>영화감상</value>
                <value>요리</value>
            </list>
        </constructor-arg>
        <property name="height" value="165" />
        <property name="weight">
            <value>45</value>
        </property>
    </bean>

    <bean id="studentInfo1" class="com.example.StudentInfo">
        <property name="student">
            <ref bean="student1" />
        </property>
    </bean>
</beans>

```

10. applicationContext2.xml

1)Namespace tab을 선택하여 c, p를 선택한다.

```

1803 <?xml version="1.0" encoding="UTF-8"?>
1804 <beans xmlns="http://www.springframework.org/schema/beans"
1805       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1806       xmlns:c="http://www.springframework.org/schema/c"
1807       xmlns:p="http://www.springframework.org/schema/p"
1808       xsi:schemaLocation="http://www.springframework.org/schema/beans
1809                          http://www.springframework.org/schema/beans/spring-beans.xsd">
1810     <bean id="student3" class="com.example.Student">
1811       <constructor-arg value="한라산" />
1812       <constructor-arg value="50" />
1813       <constructor-arg>
1814         <list>
1815           <value>노래부르기</value>
1816           <value>게임</value>
1817         </list>
1818       </constructor-arg>
1819       <property name="height" value="175" />
1820       <property name="weight">
1821         <value>75</value>
1822       </property>
1823     </bean>
1824
1825     <bean id="product" class="com.example.Product" c:pName="Computer"
1826           c:pPrice="2000000" p:maker="Samsung">
1827       <property name="color" value="Yellow" />
1828     </bean>
1829 </beans>

```

11. com.example.MainClass

```

1832 package com.example;
1833
1834 import org.springframework.context.support.AbstractApplicationContext;
1835 import org.springframework.context.support.GenericXmlApplicationContext;
1836
1837 public class MainClass {
1838     public static void main(String[] args) {
1839         String configFile = "classpath:applicationContext.xml";
1840         String configFile1 = "classpath:applicationContext2.xml";
1841         AbstractApplicationContext context = new GenericXmlApplicationContext(configFile,
1842                                         configFile1);
1843         Student student1 = context.getBean("student1", Student.class);
1844         System.out.println(student1);
1845
1846         StudentInfo studentInfo = context.getBean("studentInfo1", StudentInfo.class);
1847         Student student2 = studentInfo.getStudent();
1848         System.out.println(student2);
1849         if(student1.equals(student2)) System.out.println("Equals");
1850         else System.out.println("Different");
1851
1852         Student student3 = context.getBean("student3", Student.class);
1853         System.out.println(student3);
1854
1855         if(student1.equals(student3)) System.out.println("Equals");
1856         else System.out.println("Different");
1857
1858         Product product = context.getBean("product", Product.class);
1859         System.out.println(product);
1860         context.close();
1861     }
1862 }

```

12. Java Application 실행

```

1865 Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,weight=45.0]
1866 Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,weight=45.0]

```

1867 Equals
1868 Student [name=한라산, age=50, hobbies=[노래부르기, 게임], height=175.0,weight=75.0]
1869 Different
1870 Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]

1871
1872
1873
1874 -----

1875 Task 11. Java Annotation을 이용하여 두 개 이상의 설정 파일로 DI 설정하기

1876 1. In Package Explorer > right-click > New > Java Project

1877 1)Project Name : SpringDemo3

1878 2)JRE

1879 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

1880 3)Uncheck [Create module-info.java file]

1881 4)Next

1882 5)Finish

1883

1884

1885 2)src > right-click > New > Package

1886 1)Package name : com.example

1887 2)Finish

1888

1889

1890 3. POJO Class 생성

1891 1)com.example.Student.java

1892

1893 package com.example;

1894

1895 import java.util.List;

1896

1897 public class Student {

1898 private String name;

1899 private int age;

1900 private List<String> hobbies;

1901 private double height;

1902 private double weight;

1903 }

1904

1905 2)com.example.StudentInfo.java

1906

1907 package com.example;

1908 public class StudentInfo {

1909 private Student student;

1910 }

1911

1912 3)com.example.Product.java

1913

1914 package com.example;

1915 public class Product {

1916 private String pName;

1917 private int pPrice;

1918 private String maker;

1919 private String color;

1920 }

1921

1922

1923 4. Java Project를 Spring Project로 변환

1924 1)SpringDemo3 Project > right-click > Configure > Convert to Maven Project

1925 -Project : /SpringDemo3

1926 -Group Id : SpringDemo3

1927 -Artifact Id : SpringDemo3

1928 -version : 0.0.1-SNAPSHOT

1929 -Packaging : jar

1930 -Finish

1931

1932 2)SpringDemo3 Project > right-click > Spring > Add Spring Project Nature

1933

3)pom.xml file에 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. com.example.config package 생성

1)com.example > right-click > new > Package
2)Name : com.example.config
3)Finish

7. 2개의 Config Class 작성

1)com.example.config > right-click > New > Class
2)Name : AppConfig1
3)Finish
4)com.example.config > right-click > New > Class
5)Name : AppConfig2
6)Finish

8. Student.java, StudentInfo.java 그리고 Product.java에 lombok Annotation 붙이기

1)Student.java

```
package com.example;

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NonNull;
import lombok.RequiredArgsConstructor;

@Data
```

```

2001     @RequiredArgsConstructor
2002     @AllArgsConstructor
2003     public class Student {
2004         private @NonNull String name;
2005         private @NonNull int age;
2006         private @NonNull List<String> hobbies;
2007         private double height;
2008         private double weight;
2009     }
2010
2011 2)StudentInfo.java
2012
2013     package com.example;
2014
2015     import lombok.Setter;
2016     import lombok.Getter;
2017
2018     @Setter
2019     @Getter
2020     public class StudentInfo {
2021         private Student student;
2022     }
2023
2024 3)Product.java
2025
2026     package com.example;
2027
2028     import lombok.AllArgsConstructor;
2029     import lombok.NoArgsConstructor;
2030     import lombok.NonNull;
2031     import lombok.RequiredArgsConstructor;
2032     import lombok.Setter;
2033     import lombok.ToString;
2034
2035     @NoArgsConstructor
2036     @AllArgsConstructor
2037     @RequiredArgsConstructor
2038     @Setter
2039     @ToString
2040     public class Product {
2041         private @NonNull String pName;
2042         private @NonNull int pPrice;
2043         private @NonNull String maker;
2044         private String color;
2045     }
2046
2047
2048 9. AppConfig1.java
2049     package com.example.config;
2050
2051     import java.util.Arrays;
2052     import java.util.List;
2053
2054     import org.springframework.context.annotation.Bean;
2055     import org.springframework.context.annotation.Configuration;
2056
2057     import com.example.Student;
2058     import com.example.StudentInfo;
2059
2060     @Configuration
2061     public class AppConfig1 {
2062         @Bean
2063         public Student student1() {
2064             List<String> list = Arrays.asList("독서", "영화감상", "요리");
2065             Student student1 = new Student("백두산", 25, list);
2066             student1.setHeight(165);
2067             student1.setWeight(45);

```

```

2068         return student1;
2069     }
2070
2071     @Bean
2072     public StudentInfo studentInfo1() {
2073         StudentInfo studentInfo1 = new StudentInfo();
2074         studentInfo1.setStudent(this.student1());
2075         return studentInfo1;
2076     }
2077 }
2078
2079
2080 10. AppConfig2.java
2081     package com.example.config;
2082
2083     import java.util.Arrays;
2084     import java.util.List;
2085
2086     import org.springframework.context.annotation.Bean;
2087     import org.springframework.context.annotation.Configuration;
2088
2089     import com.example.Product;
2090     import com.example.Student;
2091
2092     @Configuration
2093     public class AppConfig2 {
2094         @Bean
2095         public Student student3() {
2096             List<String> list = Arrays.asList("노래부르기", "게임");
2097             Student student3 = new Student("한라산", 50, list);
2098             student3.setHeight(175);
2099             student3.setWeight(75);
2100             return student3;
2101         }
2102
2103         @Bean
2104         public Product product() {
2105             Product product = new Product("Computer", 2000000, "Samsung");
2106             product.setColor("Yellow");
2107             return product;
2108         }
2109     }
2110
2111
2112 11. com.example.MainClass
2113     package com.example;
2114
2115     import org.springframework.context.ApplicationContext;
2116     import org.springframework.context.annotation.AnnotationConfigApplicationContext;
2117
2118     import com.example.config.AppConfig1;
2119     import com.example.config.AppConfig2;
2120
2121     public class MainClass {
2122         public static void main(String[] args) {
2123             ApplicationContext context = new AnnotationConfigApplicationContext(AppConfig1.class,
2124                                     AppConfig2.class);
2125             Student student1 = context.getBean("student1", Student.class);
2126             System.out.println(student1);
2127
2128             StudentInfo studentInfo = context.getBean("studentInfo1", StudentInfo.class);
2129             Student student2 = studentInfo.getStudent();
2130             System.out.println(student2);
2131             if(student1.equals(student2)) System.out.println("Equals");
2132             else System.out.println("Different");
2133
2134             Student student3 = context.getBean("student3", Student.class);

```



```

2134         System.out.println(student3);
2135
2136         if(student1.equals(student3)) System.out.println("Equals");
2137         else System.out.println("Different");
2138
2139         Product product = context.getBean("product", Product.class);
2140         System.out.println(product);
2141     }
2142 }
2143
2144

```

2145 12. Java Application 실행

```

2146 Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,weight=45.0]
2147 Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,weight=45.0]
2148 Equals
2149 Student [name=한라산, age=50, hobbies=[노래부르기, 게임], height=175.0,weight=75.0]
2150 Different
2151 Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]
2152
2153
2154

```

2155 -----
 2156 Task 12. Java Annotation과 XML 을 이용한 DI 설정 방법 : XML file에 Java file을 포함시켜 사용하는 방법

2157 1. In Package Explorer > right-click > New > Java Project

```

2158 1)Project Name : SpringDemo4
2159 2)JRE
2160    -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
2161 3)Uncheck [Create module-info.java file]
2162 4)Next
2163 5)Finish
2164

```

2166 2. src > right-click > New > Package

```

2167 1)Package name : com.example
2168
2169

```

2170 3. POJO 생성

```

2171 1)com.example.Student.java
2172 package com.example;
2173
2174 import java.util.ArrayList;
2175
2176 public class Student {
2177     private String name;
2178     private int age;
2179     private ArrayList<String> hobbies;
2180     private double height;
2181     private double weight;
2182 }
2183
2184

```

2185 4. Java Project를 Spring Project로 변환

2186 1)SpringDemo4 Project > right-click > Configure > Convert to Maven Project

```

2187 -Project : /SpringDemo4
2188 -Group Id : SpringDemo4
2189 -Artifact Id : SpringDemo4
2190 -version : 0.0.1-SNAPSHOT
2191 -Packaging : jar
2192 -Finish
2193

```

2194 2)SpringDemo4 Project > right-click > Spring > Add Spring Project Nature

2196 3)pom.xml file에 Spring Context Dependency 추가하기

```

2197 <version>0.0.1-SNAPSHOT</version>
2198 <dependencies>
2199     <dependency>
2200         <groupId>org.springframework</groupId>

```

```
2201         <artifactId>spring-context</artifactId>
2202         <version>5.3.10</version>
2203     </dependency>
2204 </dependencies>
2205
```

2206 4) pom.xml > right-click > Run As > Maven install
2207 [INFO] BUILD SUCCESS 확인
2208
2209

2210 5. Lombok library 추가

- 2211 1) <https://mvnrepository.com/>에서 'lombok'으로 검색
- 2212 2) 'Project Lombok' click
- 2213 3) 1.18.20 click
- 2214 4) dependency copy해서 pom.xml에 붙여넣기

```
2215
2216     <dependencies>
2217         <dependency>
2218             <groupId>org.springframework</groupId>
2219             <artifactId>spring-context</artifactId>
2220             <version>5.3.10</version>
2221         </dependency>
2222         <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
2223         <dependency>
2224             <groupId>org.projectlombok</groupId>
2225             <artifactId>lombok</artifactId>
2226             <version>1.18.20</version>
2227             <scope>provided</scope>
2228         </dependency>
2229     </dependencies>
2230
```

2231 5) pom.xml > right-click > Run As > Maven install
2232 [INFO] BUILD SUCCESS 확인
2233
2234

2235 6. Student.java lombok Annotation 붙이기

- 2236 1) Student.java

```
2237
2238     package com.example;
2239
2240     import java.util.List;
2241
2242     import lombok.AllArgsConstructor;
2243     import lombok.Data;
2244     import lombok.NonNull;
2245     import lombok.RequiredArgsConstructor;
2246
2247     @Data
2248     @RequiredArgsConstructor
2249     @AllArgsConstructor
2250     public class Student {
2251         private @NonNull String name;
2252         private @NonNull int age;
2253         private @NonNull List<String> hobbies;
2254         private double height;
2255         private double weight;
2256     }
2257
```

2259 7. com.example.ApplicationConfig.java

```
2260     package com.example;
2261
2262     import java.util.Arrays;
2263     import java.util.List;
2264
2265     import org.springframework.context.annotation.Bean;
2266     import org.springframework.context.annotation.Configuration;
2267
```

```

2268 @Configuration
2269 public class ApplicationConfig {
2270     @Bean
2271     public Student student1(){
2272         List<String> list = Arrays.asList("독서", "영화감상", "요리");
2273
2274         Student student1 = new Student("백두산", 25, list);
2275         student1.setHeight(165);
2276         student1.setWeight(45);
2277
2278         return student1;
2279     }
2280 }

```

2283 8. SpringDemo4/resources folder 생성

- 2284 1)SpringDemo4 project > right-click > Build Path > Configure Build Path
- 2285 2)Source Tab > Add Folder
- 2286 3)SpringDemo4 선택 확인
- 2287 4)Create New Folder > Folder name : resources > Finish > OK
- 2288 5)SpringDemo4/resources(new) 확인
- 2289 6)Apply and Close

2292 9. Bean Configuration XML 작성

- 2293 1)SpringDemo4/resources > right-click > New > Spring Bean Configuration File
- 2294 2)File name : applicationContext.xml > Finish

2297 10. /resources/applicationContext.xml

```

2298 <?xml version="1.0" encoding="UTF-8"?>
2299 <beans xmlns="http://www.springframework.org/schema/beans"
2300     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2301     xmlns:context="http://www.springframework.org/schema/context"
2302     xsi:schemaLocation="http://www.springframework.org/schema/beans
2303         http://www.springframework.org/schema/beans/spring-beans.xsd">
2304
2305     <bean class="org.springframework.context.annotation.ConfigurationClassPostProcessor" />
2306     <bean class="com.example.ApplicationConfig" />
2307     <bean id="student3" class="com.example.Student">
2308         <constructor-arg value="북한산" />
2309         <constructor-arg value="50" />
2310         <constructor-arg>
2311             <list>
2312                 <value>노래부르기</value>
2313                 <value>게임</value>
2314             </list>
2315         </constructor-arg>
2316         <property name="height" value="175" />
2317         <property name="weight">
2318             <value>75</value>
2319         </property>
2320     </bean>
2321 </beans>

```

2323 11. com.example.MainClass.java

```

2324 package com.example;
2325
2326 import org.springframework.context.support.AbstractApplicationContext;
2327 import org.springframework.context.support.GenericXmlApplicationContext;
2328
2329 public class MainClass {
2330     public static void main(String[] args) {
2331         String configFile = "classpath:applicationContext.xml";
2332         AbstractApplicationContext context = new GenericXmlApplicationContext(configFile);
2333         Student student1 = context.getBean("student1", Student.class);

```

```
2334         System.out.println(student1);
2335
2336         Student student3 = context.getBean("student3", Student.class);
2337         System.out.println(student3);
2338     }
2339 }
2340
2341
```

2342 12. Java Application 실행

```
2343     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,weight=45.0]
2344     Student [name=북한산, age=50, hobbies=[노래부르기, 게임], height=175.0,weight=75.0]
```

2345

2346

2347 13. JUnit을 사용한 DI test class 작성하기

2348 1)com.example > right-click > New > JUnit Test Case

2349 2)Select [New JUnit 4 test]

2350 3)Name : HelloBeanJUnitTest

2351 4)Finish

2352 5)[New JUnit Test Case] 창에서 Select [Perform the following action:] > Add JUnit 4 library to the build path

2353 6)OK

2354

2355

2356 14. JUnit을 사용한 Test

2357 1)src/com.example > New > Class

2358 -Name : HelloBeanJUnitTest.java

2359

2360 package com.example;

2361

2362 import static org.junit.Assert.assertEquals;

2363 import static org.junit.Assert.assertSame;

2364

2365 import org.junit.Before;

2366 import org.junit.Test;

2367 import org.springframework.context.ApplicationContext;

2368 import org.springframework.context.support.GenericXmlApplicationContext;

2369

2370 public class HelloBeanJUnitTest {

2371 private ApplicationContext context;

2372

2373 @Before

2374 public void init(){

2375 context = new GenericXmlApplicationContext("classpath:applicationContext.xml");

2376 }

2377

2378 @Test

2379 public void test1(){

2380 Student student1 = (Student)context.getBean("student1");

2381 assertEquals("백두산", student1.getName());

2382 System.out.println(student1);

2383 }

2384

2385 @Test

2386 public void test2(){

2387 Student student3 = context.getBean("student3", Student.class);

2388 System.out.println(student3);

2389

2390 Student student4 = (Student)context.getBean("student3");

2391 assertEquals(student3, student4);

2392 }

2393 }

2394

2395 2)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test

2396 -JUnit 창에 Green Bar

2397 Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0, weight=45.0)

2398 Student(name=북한산, age=50, hobbies=[노래부르기, 게임], height=175.0, weight=75.0)

2399

15. Spring TestContext Framework을 이용한 Test

1)Spring-Test library 설치

- <http://mvnrepository.com>에서 'spring test'로 검색
- 검색 결과 목록에서 'Spring TestContext Framework' Click
- version 목록에서 5.3.10 Click

2)dependency 복사해서 pom.xml에 붙여넣기

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-test</artifactId>
  <version>5.3.10</version>
  <scope>test</scope>
</dependency>
```

3)pom.xml > right-click > Maven Install

[INFO] BUILD SUCCESS

4)Spring-Test를 사용할 HelloBeanJUnitSpringTest.java 작성

```
-src/com.example > New > Class
-Name : HelloBeanJUnitSpringTest
-Finish

package com.example;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertSame;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations="classpath:applicationContext.xml")
public class HelloBeanJUnitSpringTest {
    @Autowired
    private ApplicationContext context;

    @Test
    public void test1() {
        Student student1 = this.context.getBean("student1", Student.class);
        assertEquals(25, student1.getAge());
        System.out.println(student1);
    }

    @Test
    public void test2() {
        Student student3 = (Student)this.context.getBean("student3");
        Student student4 = this.context.getBean("student3", Student.class);
        assertEquals(student3, student4);
        System.out.println(student4);
    }
}
```

-right-click > Run As > JUnit Test

-결과 -> JUnit View에 초록색 bar

5)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면

- 해당 Project > right-click > Build Path > Libraries tab
- spring-test-5.3.10.jar 선택 후 [Remove] 로 삭제
- Classpath 선택
- [Add External JARs...] Click
- Local M2 Repository(e.g

C:\Users\instructor\.m2\repository\org\springframework\spring-test\5.3.10)에서 직접
jar(spring-test-5.3.10.jar)를 선택할 것

-[Order and Export] tab에서 spring-test-5.3.10.jar 선택 후 [Up] button을 클릭
-해당 DIdemo/src 바로 아래까지 올리고 [Apply and Close] Click

Task 13. Java Annotation과 XML 을 이용한 DI 설정 방법 : Java file에 XML file을 포함시켜 사용하는 방법

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

1)Project Name : SpringDemo5

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

2)Finish

3. com.example.Student.java

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)SpringDemo5 Project > right-click > Configure > Convert to Maven Project

-Project : /SpringDemo5

-Group Id : SpringDemo5

-Artifact Id : SpringDemo5

-version : 0.0.1-SNAPSHOT

-Packaging : jar

-Finish

2)SpringDemo5 Project > right-click > Spring > Add Spring Project Nature

3)pom.xml file에 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

```

2532
2533     <dependencies>
2534         <dependency>
2535             <groupId>org.springframework</groupId>
2536             <artifactId>spring-context</artifactId>
2537             <version>5.3.10</version>
2538         </dependency>
2539         <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
2540         <dependency>
2541             <groupId>org.projectlombok</groupId>
2542             <artifactId>lombok</artifactId>
2543             <version>1.18.20</version>
2544             <scope>provided</scope>
2545         </dependency>
2546     </dependencies>
2547
2548 5)pom.xml > right-click > Run As > Maven install
2549 [INFO] BUILD SUCCESS 확인
2550
2551 6. Student.java lombok Annotation 붙이기
2552 1)Student.java
2553
2554     package com.example;
2555
2556     import java.util.List;
2557
2558     import lombok.AllArgsConstructor;
2559     import lombok.Data;
2560     import lombok.NonNull;
2561     import lombok.RequiredArgsConstructor;
2562
2563     @Data
2564     @RequiredArgsConstructor
2565     @AllArgsConstructor
2566     public class Student {
2567         private @NonNull String name;
2568         private @NonNull int age;
2569         private @NonNull List<String> hobbies;
2570         private double height;
2571         private double weight;
2572     }
2573
2574
2575 7. SpringDemo5/resources folder 생성
2576 1)SpringDemo5 project > right-click > Build Path > Configure Build Path
2577 2)Source Tab > Add Folder
2578 3)SpringDemo5 선택 확인
2579 4>Create New Folder > Folder name : resources > Finish > OK
2580 5)SpringDemo5/resources(new) 확인
2581 6)Apply and Close
2582
2583
2584 8. Bean Configuration XML 작성
2585 1)SpringDemo5/resources > right-click > New > Spring Bean Configuration File
2586 2)File name : applicationContext.xml > Finish
2587
2588
2589 9. /resources/applicationContext.xml
2590 <?xml version="1.0" encoding="UTF-8"?>
2591 <beans xmlns="http://www.springframework.org/schema/beans"
2592     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2593     xsi:schemaLocation="http://www.springframework.org/schema/beans
2594     http://www.springframework.org/schema/beans/spring-beans.xsd">
2595
2596     <bean id="student3" class="com.example.Student">
2597         <constructor-arg value="지리산" />

```

```

2598         <constructor-arg value="30" />
2599         <constructor-arg>
2600             <list>
2601                 <value>등산</value>
2602                 <value>게임</value>
2603                 <value>독서</value>
2604             </list>
2605         </constructor-arg>
2606         <property name="height" value="165" />
2607         <property name="weight">
2608             <value>49</value>
2609         </property>
2610     </bean>
2611 </beans>

```

```

2612
2613
2614 10. com.example.ApplicationConfig.java
2615     package com.example;

```

```

2616
2617     import java.util.Arrays;
2618     import java.util.List;
2619
2620     import org.springframework.context.annotation.Bean;
2621     import org.springframework.context.annotation.Configuration;
2622     import org.springframework.context.annotation.ImportResource;
2623
2624     @Configuration
2625     @ImportResource("classpath:ApplicationContext.xml")
2626     public class ApplicationConfig {
2627         @Bean
2628         public Student student1(){
2629             List<String> hobbies = Arrays.asList("독서", "영화감상", "요리");
2630
2631             Student student = new Student("백두산", 25, hobbies);
2632             student.setHeight(165);
2633             student.setWeight(45);
2634
2635             return student;
2636         }
2637     }

```

```

2638
2639
2640 11. com.example.MainClass.java
2641     package com.example;
2642
2643     import org.springframework.context.annotation.AnnotationConfigApplicationContext;
2644
2645     public class MainClass {
2646         public static void main(String[] args) {
2647             AnnotationConfigApplicationContext context = new
                AnnotationConfigApplicationContext(ApplicationConfig.class);
2648             Student student1 = context.getBean("student1", Student.class);
2649             System.out.println(student1);
2650
2651             Student student3 = context.getBean("student3", Student.class);
2652             System.out.println(student3);
2653
2654             context.close();
2655         }
2656     }

```

```

2657
2658
2659 12. Java Application 실행
2660     Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0, weight=45.0)
2661     Student(name=지리산, age=30, hobbies=[등산, 게임, 독서], height=165.0, weight=49.0)

```

```

2662
2663

```



```
2664 13. JUnit 5를 사용한 DI test class 작성하기
2665     1)com.example > right-click > New > JUnit Test Case
2666     2)Select [New JUnit Jupiter test]
2667     3)Name : HelloBeanJUnitTest
2668     4)Finish
2669     5)[New JUnit Test Case] 창에서 Select [Perform the following action:] > Add JUnit 5 library to the
    build path
2670     6)OK
2671
2672
2673 14. pom.xml에 dependency 추가
2674     1)JUnit 5 설치
2675         -http://mvnrepository.com에서 'junit'로 검색
2676         -검색 결과 목록에서 'JUnit Jupiter API' Click
2677         -version 목록에서 5.8.1 click
2678
2679     2)dependency 복사해서 pom.xml에 붙여넣기
2680         <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
2681         <dependency>
2682             <groupId>org.junit.jupiter</groupId>
2683             <artifactId>junit-jupiter-api</artifactId>
2684             <version>5.8.1</version>
2685             <scope>test</scope>
2686         </dependency>
2687
2688     3)pom.xml > right-click > Maven Install
2689         [INFO] BUILD SUCCESS
2690         -만일 ERROR 발생하면 다음과 같이 조치한다.
2691         -SpringDemo5 > right-click > Maven > Update Project
2692         -SpringDemo5가 check되어 있음을 확인하고 OK
2693         -다시 pom.xml > right-click > Maven Install
2694         [INFO] BUILD SUCCESS
2695
2696
2697 15. JUnit 5를 사용한 Test
2698     1)com.example.HelloBeanJUnitTest.java
2699
2700         package com.example;
2701
2702         import static org.junit.jupiter.api.Assertions.assertEquals;
2703         import static org.junit.jupiter.api.Assertions.assertSame;
2704
2705         import org.junit.jupiter.api.BeforeEach;
2706         import org.junit.jupiter.api.Test;
2707         import org.springframework.context.ApplicationContext;
2708         import org.springframework.context.annotation.AnnotationConfigApplicationContext;
2709
2710         class HelloBeanJUnitTest {
2711             private ApplicationContext context;
2712
2713             @BeforeEach
2714             public void init() {
2715                 this.context = new AnnotationConfigApplicationContext(ApplicationConfig.class);
2716             }
2717
2718             @Test
2719             public void test1(){
2720                 Student student1 = (Student)context.getBean("student1");
2721                 assertEquals("백두산", student1.getName());
2722                 System.out.println(student1);
2723             }
2724
2725             @Test
2726             public void test2() {
2727                 Student student3 = context.getBean("student3", Student.class);
2728                 Student student4 = (Student)context.getBean("student3");
2729                 assertSame(student3, student4);
2730             }
2731         }
```

```

2730         System.out.println(student3);
2731     }
2732 }
2733
2734 2)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test
2735     -JUnit 창에 Green Bar
2736     Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0, weight=45.0)
2737     Student(name=홍지민, age=30, hobbies=[등산, 게임, 독서], height=165.0, weight=49.0)
2738
2739
2740
2741 -----
2742 Task 14. Lab
2743 1. In Package Explorer > right-click > New > Java Project
2744     1)Project name : DIDemo3
2745     2)JRE
2746         -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
2747     3)Uncheck [Create module-info.java file]
2748     4)Next
2749     5)Finish
2750
2751
2752 2. src > right-click > New > Package
2753     1)Package name : com.example
2754     2)Finish
2755
2756
2757 3. POJO class 작성
2758     1)com.example > right-click > New > Class
2759     2)Class Name : Hello
2760
2761     package com.example;
2762
2763     public class Hello{
2764         private String name;
2765         private Printer printer;
2766
2767         public String sayHello(){
2768             return "Hello " + name;
2769         }
2770
2771         public void print(){
2772             this.printer.print(sayHello());
2773         }
2774     }
2775
2776 3)com.example > right-click > New > Interface
2777 4)interface name : Printer
2778
2779     package com.example;
2780
2781     public interface Printer{
2782         void print(String message);
2783     }
2784
2785 5)com.example > right-click > New > Class
2786 6)Class Name : StringPrinter
2787 7)Interfaces : com.example.Printer
2788
2789     package com.example;
2790
2791     public class StringPrinter implements Printer{
2792         private StringBuffer buffer = new StringBuffer();
2793
2794         @Override
2795         public void print(String message){
2796             this.buffer.append(message);

```

```
2797     }
2798
2799     public String toString(){
2800         return this.buffer.toString();
2801     }
2802 }
2803
```

2804 8)com.example > right-click > New > Class

2805 9)Class Name : ConsolePrinter

2806 10)Interfaces : com.example.Printer

```
2807
2808     package com.example;
2809
2810     public class ConsolePrinter implements Printer{
2811
2812         @Override
2813         public void print(String message){
2814             System.out.println(message);
2815         }
2816     }
2817
2818
```

2819 4. Java Project를 Spring Project로 변환

2820 1)DIDemo3 Project > right-click > Configure > Convert to Maven Project

2821 -Project : /DIDemo3

2822 -Group Id : DIDemo3

2823 -Artifact Id : DIDemo3

2824 -version : 0.0.1-SNAPSHOT

2825 -Packaging : jar

2826 -Finish

2827

2828 2)DIDemo3 Project > right-click > Spring > Add Spring Project Nature

2829

2830 3)pom.xml file에 Spring Context Dependency 추가하기

2831 <version>0.0.1-SNAPSHOT</version>

2832 <dependencies>

2833 <dependency>

2834 <groupId>org.springframework</groupId>

2835 <artifactId>spring-context</artifactId>

2836 <version>5.3.10</version>

2837 </dependency>

2838 </dependencies>

2839

2840 4)pom.xml > right-click > Run As > Maven install

2841 [INFO] BUILD SUCCESS 확인

2842

2843

2844 5. Lombok library 추가

2845 1)<https://mvnrepository.com/>에서 'lombok'으로 검색

2846 2)'Project Lombok' click

2847 3)1.18.20 click

2848 4)dependency copy해서 pom.xml에 붙여넣기

2849

2850 <dependencies>

2851 <dependency>

2852 <groupId>org.springframework</groupId>

2853 <artifactId>spring-context</artifactId>

2854 <version>5.3.10</version>

2855 </dependency>

2856 <!-- <https://mvnrepository.com/artifact/org.projectlombok/lombok> -->

2857 <dependency>

2858 <groupId>org.projectlombok</groupId>

2859 <artifactId>lombok</artifactId>

2860 <version>1.18.20</version>

2861 <scope>provided</scope>

2862 </dependency>

2863 </dependencies>

2864
2865 5)pom.xml > right-click > Run As > Maven install
2866 [INFO] BUILD SUCCESS 확인
2867

2868
2869 6. Hello.java에 lombok Annotation으로 수정하기
2870

```
2871 package com.example;  
2872  
2873 import lombok.NoArgsConstructor;  
2874 import lombok.Setter;  
2875  
2876 @Setter  
2877 @NoArgsConstructor  
2878 public class Hello {  
2879     private String name;  
2880     private Printer printer;  
2881  
2882     public String sayHello(){  
2883         return "Hello " + name;  
2884     }  
2885  
2886     public void print(){  
2887         this.printer.print(sayHello());  
2888     }  
2889 }  
2890  
2891
```

2892 7. src/config folder 생성

- 2893 1)/src > right-click > New > Folder
- 2894 2)Folder name : config
2895

2896
2897 8. Bean Configuration XML 작성

- 2898 1)/src/config > right-click > New > Other > Spring > Spring Bean Configuration File
- 2899 2)File name : beans.xml > Finish
2900

```
2901 <?xml version="1.0" encoding="UTF-8"?>  
2902 <beans xmlns="http://www.springframework.org/schema/beans"  
2903     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
2904     xsi:schemaLocation="http://www.springframework.org/schema/beans  
http://www.springframework.org/schema/beans/spring-beans.xsd">  
2905  
2906     <bean id="hello" class="com.example.Hello">  
2907         <property name="name" value="Spring" />  
2908         <property name="printer" ref="printer" />  
2909     </bean>  
2910     <bean id="printer" class="com.example.StringPrinter" />  
2911     <bean id="consolePrinter" class="com.example.ConsolePrinter" />  
2912 </beans>  
2913  
2914
```

2915 9. DI Test class 작성

- 2916 1)/src > right-click > New > Package
- 2917 2)Name : com.example.test
- 2918 3)Finish
- 2919 4)/src/com.example.test > right-click > New > Class
- 2920 5)Name : HelloBeanTest
2921

```
2922 package com.example.test;  
2923  
2924 import org.springframework.context.ApplicationContext;  
2925 import org.springframework.context.support.GenericXmlApplicationContext;  
2926  
2927 import com.example.Hello;  
2928 import com.example.Printer;  
2929
```

```

2930 public class HelloBeanTest {
2931     public static void main(String [] args){
2932         ApplicationContext context = new GenericXmlApplicationContext("config/beans.xml");
2933
2934         Hello hello = (Hello)context.getBean("hello");
2935         System.out.println(hello.sayHello());
2936         hello.print();
2937
2938         Printer printer = (Printer)context.getBean("printer");
2939         System.out.println(printer.toString());
2940
2941         Hello hello2 = context.getBean("hello", Hello.class);
2942         hello2.print();
2943
2944         System.out.println(hello == hello2); //Singleton Pattern
2945     }
2946 }

```

6)Java Application 실행

Hello Spring

Hello Spring

true

10. JUnit 5 Library 설치

1)JUnit 5 설치

-<http://mvnrepository.com>에서 'junit'로 검색

-검색 결과 목록에서 'JUnit Jupiter API' Click

-version 목록에서 5.8.1 click

2)dependency 복사해서 pom.xml에 붙여넣기

```

2961 <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
2962 <dependency>
2963     <groupId>org.junit.jupiter</groupId>
2964     <artifactId>junit-jupiter-api</artifactId>
2965     <version>5.8.1</version>
2966     <scope>test</scope>
2967 </dependency>

```

3)pom.xml > right-click > Maven Install

[INFO] BUILD SUCCESS

-만일 ERROR 발생하면 다음과 같이 조치한다.

-SpringDemo5 > right-click > Maven > Update Project

-SpringDemo5가 check되어 있음을 확인하고 OK

-다시 pom.xml > right-click > Maven Install

[INFO] BUILD SUCCESS

11. JUnit 5를 사용한 Test

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

2)Name : HelloBeanJUnitTest

```

2982 package com.example.test;

```

```

2984 import static org.junit.jupiter.api.Assertions.assertEquals;

```

```

2985 import static org.junit.jupiter.api.Assertions.assertSame;

```

```

2987 import org.junit.jupiter.api.BeforeEach;

```

```

2988 import org.junit.jupiter.api.Test;

```

```

2989 import org.springframework.context.ApplicationContext;

```

```

2990 import org.springframework.context.support.GenericXmlApplicationContext;

```

```

2992 import com.example.Hello;

```

```

2994 public class HelloBeanJUnitTest {

```

```

2995     private ApplicationContext context;

```

```

2996

```

```

2997     @BeforeEach
2998     public void init() {
2999         this.context = new GenericXmlApplicationContext("config/beans.xml");
3000     }
3001
3002     @Test
3003     public void test1(){
3004         Hello hello = (Hello)context.getBean("hello");
3005         assertEquals("Hello Spring", hello.sayHello());
3006         hello.print();
3007     }
3008
3009     @Test
3010     public void test2(){
3011         Hello hello = (Hello)context.getBean("hello");
3012         Hello hello2 = context.getBean("hello", Hello.class);
3013         assertSame(hello, hello2);
3014     }
3015 }

```

3)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면

- 해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
- Classpath 선택
- [Add External JARs...] Click
- Local M2 Repository(e.g C:\Users\사용자아이디\.m2\repository\org\junit\jupiter\junit-jupiter-api\5.8.1)에서 직접 jar(junit-jupiter-api-5.8.1.jar)를 선택할 것
- [Order and Export] tab에서 junit-jupiter-api-5.8.1.jar 선택 후 [Up] button을 클릭
- 해당 Project/src 바로 아래까지 올리고 [Apply and Close] Click

4)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test

- JUnit 창에 Green Bar

12. Spring TestContext Framework

1)Spring-Test library 설치

2)pom.xml 수정

```

<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-test</artifactId>
  <version>5.3.10</version>
  <scope>test</scope>
</dependency>

```

3)pom.xml > right-click > Maven Install

- 만일 Error 발생시 DIDemo3 > right-click > Maven > Update Project... > Ok
- 다시 Maven Install 실행

4)Spring-Test를 사용할 DI test class-HelloBeanJUnitSpringTest.java 작성하기

- /src/com.example.test > New > Class
- Name : HelloBeanJUnitSpringTest
- Finish

```

package com.example.test;

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

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.extension.ExtendWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit.jupiter.SpringExtension;

```

```

3062     import com.example.Hello;
3063
3064     @ExtendWith(SpringExtension.class)
3065     //JUnit 5.x에서 사용
3066     @ContextConfiguration(locations="classpath:config/beans.xml")
3067     public class HelloBeanJUnitSpringTest {
3068         @Autowired
3069         ApplicationContext context;
3070
3071         @Test
3072         public void test1(){
3073             Hello hello = (Hello)context.getBean("hello");
3074             assertEquals("Hello Spring", hello.sayHello());
3075             hello.print();
3076         }
3077
3078         @Test
3079         public void test2(){
3080             Hello hello = (Hello)context.getBean("hello");
3081             Hello hello2 = context.getBean("hello", Hello.class);
3082             assertSame(hello, hello2);
3083         }
3084     }
3085
3086 5)right-click > Run As > Junit Test
3087 6)결과 -> Junit View에 초록색 bar
3088 7)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
3089     -해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
3090     -spring-test-5.3.10.jar 선택 후 [Remove] 로 삭제
3091     -Classpath 선택
3092     -[Add External JARs...] Click
3093     -Local M2 Repository(e.g
3094       C:\Users\사용자아이디\.m2\repository\org\springframework\spring-test\5.3.10)에서 직접
3095       jar(spring-test-5.3.10.jar)를 선택할 것
3096     -[Order and Export] tab에서 spring-test-5.3.10.jar 선택 후 [Up] button을 클릭
3097     -해당 Project/src 바로 아래까지 올리고 [Apply and Close] Click
3098
3098 13. src/com.example/StringPrinter.java 수정
3099     package com.example;
3100
3101     import org.springframework.stereotype.Component;
3102
3103     @Component("stringPrinter")
3104     public class StringPrinter implements Printer{
3105         private StringBuffer buffer = new StringBuffer();
3106         ...
3107
3108
3109 14. src/com.example/ConsolePrinter.java 수정
3110     package com.example;
3111
3112     import org.springframework.stereotype.Component;
3113
3114     @Component("consolePrinter")
3115     public class ConsolePrinter implements Printer{
3116         ...
3117
3118
3119
3120 15. /src/com.example/Hello.java 수정
3121     package com.example;
3122
3123     //import org.springframework.beans.factory.annotation.Autowired;
3124     //import org.springframework.beans.factory.annotation.Qualifier;
3125     import org.springframework.beans.factory.annotation.Value;
3126     import org.springframework.stereotype.Component;

```

```

3127 //import javax.annotation.Resource;
3128
3129 import javax.inject.Inject;
3130 import javax.inject.Named;
3131
3132 import lombok.NoArgsConstructor;
3133 import lombok.Setter;
3134
3135 @Setter
3136 @NoArgsConstructor
3137 @Component
3138 public class Hello {
3139     @Value("Spring")
3140     private String name;
3141
3142     //@Autowired(required = true)
3143     //@Qualifier("stringPrinter")
3144     //@Resource(name = "stringPrinter")
3145     /*
3146     @Resource annotation 사용하려면 mvnrepository.com에서 Javax Annotation API 검색해서
3147     <dependency>
3148     <groupId>javax.annotation</groupId>
3149     <artifactId>javax.annotation-api</artifactId>
3150     <version>1.3.2</version>
3151     </dependency>
3152
3153     @Inject annotation 사용하려면 mvnrepository.com에서 Javax Inject API 검색해서
3154     <dependency>
3155     <groupId>javax.inject</groupId>
3156     <artifactId>javax.Inject</artifactId>
3157     <version>1</version>
3158     </dependency>
3159     */
3160     @Inject
3161     @Named("stringPrinter")
3162     private Printer printer;
3163
3164     public String sayHello(){
3165         return "Hello " + name;
3166     }
3167
3168     public void print(){
3169         this.printer.print(sayHello());
3170     }
3171 }
3172
3173

```

3174 16. 기존의 설정file과 충돌이 발생하기 때문에 /src/config/beans.xml 삭제

3175

3176

3177 17. 새로운 설정 file 생성

3178 1)src/config > right-click > New > Other > Spring > Spring Bean Configuration File

3179 2)File name : annos.xml > Finish

3180 3)Namespace tab > context Check

```

3181
3182 <?xml version="1.0" encoding="UTF-8"?>
3183 <beans xmlns="http://www.springframework.org/schema/beans"
3184     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3185     xmlns:context="http://www.springframework.org/schema/context"
3186     xsi:schemaLocation="http://www.springframework.org/schema/beans
3187         http://www.springframework.org/schema/beans/spring-beans.xsd
3188         http://www.springframework.org/schema/context
3189         http://www.springframework.org/schema/context/spring-context-4.3.xsd">
3188
3189     <context:component-scan base-package="com.example" />
3190 </beans>
3191

```



```

3192
3193 18. /src/com.example.test/HelloBeanJUnitSpringTest.java 수정하기
3194     package com.example.test;
3195
3196     import static org.junit.jupiter.api.Assertions.assertEquals;
3197     import static org.junit.jupiter.api.Assertions.assertSame;
3198
3199     import org.junit.jupiter.api.Test;
3200     import org.junit.jupiter.api.extension.ExtendWith;
3201     import org.springframework.beans.factory.annotation.Autowired;
3202     import org.springframework.context.ApplicationContext;
3203     import org.springframework.test.context.ContextConfiguration;
3204     import org.springframework.test.context.junit.jupiter.SpringExtension;
3205
3206     import com.example.Hello;
3207
3208     @ExtendWith(SpringExtension.class)
3209     @ContextConfiguration(locations="classpath:config/annos.xml")
3210     public class HelloBeanJUnitSpringTest {
3211         @Autowired
3212         ApplicationContext context;
3213
3214         @Test
3215         public void test1(){
3216             Hello hello = (Hello)context.getBean("hello");
3217             assertEquals("Hello Spring", hello.sayHello());
3218             hello.print();
3219         }
3220
3221         @Test
3222         public void test2(){
3223             Hello hello = (Hello)context.getBean("hello");
3224             Hello hello2 = context.getBean("hello", Hello.class);
3225             assertSame(hello, hello2);
3226         }
3227     }
3228

```

- 1)right-click > Run As > Junit Test
- 2)결과 -> Junit View에 초록색 bar

```

3231
3232
3233
3234 -----
3235 Task 15. Lab with JUnit 5 Jupiter
3236 1. In Package Explorer > right-click > New > Java Project
3237     1)Project Name : DIDemo4
3238     2)JRE
3239         -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
3240     3)Uncheck [Create module-info.java file]
3241     4)Next
3242     5)Finish
3243
3244
3245 2. src > right-click > New > Package
3246     1)Package name : com.example
3247     2)Finish
3248
3249
3250 3. com.example.Student.java, com.example.StudentInfo.java
3251     1)Student.java
3252         package com.example;
3253
3254         import java.util.List;
3255
3256         public class Student {
3257             private String name;
3258             private int age;

```

```
3259     private List<String> hobbies;  
3260     private double height;  
3261     private double weight;  
3262 }  
3263
```

2) StudentInfo.java

```
package com.example;  
  
public class StudentInfo {  
    private Student student;  
}
```

4. Java Project를 Spring Project로 변환

1) DIDemo4 Project > right-click > Configure > Convert to Maven Project

```
-Project : /DIDemo4  
-Group Id : DIDemo4  
-Artifact Id : DIDemo4  
-version : 0.0.1-SNAPSHOT  
-Packaging : jar  
-Finish
```

2) DIDemo4 Project > right-click > Spring > Add Spring Project Nature

3) pom.xml file에 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, StudentInfo.java lombok Annotation 붙이기

1) Student.java

```
package com.example;
```

```

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NonNull;
import lombok.RequiredArgsConstructor;

@Data
@RequiredArgsConstructor
@AllArgsConstructor
public class Student {
    private @NonNull String name;
    private @NonNull int age;
    private @NonNull List<String> hobbies;
    private double height;
    private double weight;
}

```

2)StudentInfo.java

```

package com.example;

import lombok.AllArgsConstructor;
import lombok.Setter;

@Setter
@AllArgsConstructor
public class StudentInfo {
    private Student student;

    public void printInfo(){
        if(this.student != null){
            System.out.println("Name : " + this.student.getName());
            System.out.println("Age : " + this.student.getAge());
            System.out.println("Hobbies");
            this.student.getHobbys().forEach(hobby -> System.out.println(hobby));
            System.out.println("Height : " + this.student.getHeight());
            System.out.println("Weight : " + this.student.getWeight());
        }
    }
}

```

7. com.example.config package 생성

- 1)com.example > right-click > New > Package
- 2)Name : com.example.config
- 3)Finish

8. com.example.config.ApplicationConfig.java 생성

- 1)com.example.config > right-click > New > Click
- 2)Name : ApplicationConfig
- 3)Finish

```

package com.example.config;

import java.util.Arrays;
import java.util.List;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import com.example.Student;
import com.example.StudentInfo;

@Configuration

```

```

3393 public class ApplicationConfig {
3394     @Bean
3395     public Student student1() {
3396         List<String> list = Arrays.asList("독서", "영화감상", "요리");
3397         Student student1 = new Student("백두산", 25, list);
3398         student1.setHeight(165);
3399         student1.setWeight(45);
3400         return student1;
3401     }
3402
3403     @Bean
3404     public StudentInfo studentInfo() {
3405         return new StudentInfo(this.student1());
3406     }
3407 }
3408
3409

```

3410 9. com.example.MainClass.java

```

3411 package com.example;
3412
3413 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
3414
3415 import com.example.config.ApplicationConfig;
3416
3417 public class MainClass {
3418     public static void main(String[] args) {
3419         AnnotationConfigApplicationContext context = new
3420             AnnotationConfigApplicationContext(ApplicationConfig.class);
3421         Student student1 = context.getBean("student1", Student.class);
3422         System.out.println(student1);
3423
3424         StudentInfo studentInfo = context.getBean("studentInfo", StudentInfo.class);
3425         studentInfo.setStudent(student1);
3426         studentInfo.printInfo();
3427
3428         context.close();
3429     }
3430 }
3431
3432

```

3433 10. Java Application 실행

```

3434 Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0, weight=45.0)
3435 Name : 백두산
3436 Age : 25
3437 Hobbies
3438 독서
3439 영화감상
3440 요리
3441 Height : 165.0
3442 Weight : 45.0
3443
3444

```

3445 11. Student.java 수정

```

3446 package com.example;
3447
3448 import java.util.List;
3449
3450 import org.springframework.beans.factory.annotation.Value;
3451 import org.springframework.stereotype.Component;
3452
3453 import lombok.Getter;
3454 import lombok.Setter;
3455
3456 @Component
3457 @Setter

```

```

3459 @Getter
3460 public class Student {
3461     @Value("백두산")
3462     private String name;
3463     @Value("25")
3464     private int age;
3465     @Value("등산, 게임, 독서")
3466     private List<String> hobbies;
3467     @Value("162.5")
3468     private double height;
3469     @Value("49.2")
3470     private double weight;
3471 }
3472
3473

```

3474 12. StudentInfo.java 수정

```

3475 package com.example;
3476
3477 import org.springframework.beans.factory.annotation.Autowired;
3478 import org.springframework.stereotype.Component;
3479
3480 import lombok.NoArgsConstructor;
3481 import lombok.Setter;
3482
3483 @NoArgsConstructor
3484 @Component
3485 public class StudentInfo {
3486     @Setter(onMethod_ = @Autowired)
3487     private Student student;
3488
3489     public void printInfo(){
3490         if(this.student != null){
3491             System.out.println("Name : " + this.student.getName());
3492             System.out.println("Age : " + this.student.getAge());
3493             System.out.println("Hobbies");
3494             this.student.getHobbies().forEach(hobby -> System.out.println(hobby));
3495             System.out.println("Height : " + this.student.getHeight());
3496             System.out.println("Weight : " + this.student.getWeight());
3497         }else {
3498             System.out.println("Null");
3499         }
3500     }
3501 }
3502
3503
3504

```

3505 13. ApplicationConfig.java 수정

```

3506 package com.example.config;
3507
3508 import org.springframework.context.annotation.Bean;
3509 import org.springframework.context.annotation.ComponentScan;
3510 import org.springframework.context.annotation.Configuration;
3511
3512 import com.example.StudentInfo;
3513
3514 @Configuration
3515 @ComponentScan(basePackages = {"com.example"})
3516 public class ApplicationConfig {
3517     @Bean
3518     public StudentInfo studentInfo() {
3519         return new StudentInfo();
3520     }
3521 }
3522
3523
3524

```

3525 14. MainClass.java 수정

```
3526
3527 package com.example;
3528
3529 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
3530
3531 import com.example.config.ApplicationConfig;
3532
3533 public class MainClass {
3534     public static void main(String[] args) {
3535         AnnotationConfigApplicationContext context = new
            AnnotationConfigApplicationContext(ApplicationConfig.class);
3536         StudentInfo info = context.getBean("studentInfo", StudentInfo.class);
3537         info.printInfo();
3538         context.close();
3539     }
3540 }
```

```
3541
3542
3543 15. MainClass 실행
```

```
3544
3545 Name : 백두산
3546 Age : 25
3547 Hobbies
3548 등산, 게임, 독서
3549 Height : 162.5
3550 Weight : 49.2
```