

```
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

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

111

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

135 3) pom.xml file에 Spring Context Dependency 추가하기

136 -<https://mvnrepository.com>에서 'spring context'로 검색

137 -[Spring Context] click

138 -현재 Spring 5.x의 현재 version인 5.3.10 click

139 -Copy하여 pom.xml에 paste

140

141 <version>0.0.1-SNAPSHOT</version>

142 <dependencies> <--- dependencies element 추가

143 <dependency> <---여기에 paste

144 <groupId>org.springframework</groupId>

145 <artifactId>spring-context</artifactId>

146 <version>5.3.10</version>

147 </dependency>

148 </dependencies>

149

150 4) pom.xml Save

151

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

153 [INFO] BUILD SUCCESS 확인

154

155

156 6. config folder 생성

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

158 -Folder name : config

159 -Finish

160

161

162 7. Bean Configuration XML 작성

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

164 2) Name : applicationContext.xml > Finish

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

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

```

167     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
168     xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
169
170     <bean id="calculator" class="com.example.Calculator" />
171
172     <bean id="myCalculator" class="com.example.MyCalculator">
173         <property name="calculator">
174             <ref bean="calculator" />
175         </property>
176         <property name="firstNum" value="10" />
177         <property name="secondNum" value="2" />
178     </bean>
179 </beans>

```

8. MainClass.java

```

183 package com.javasoft;
184
185 import org.springframework.context.support.AbstractApplicationContext;
186 import org.springframework.context.support.GenericXmlApplicationContext;
187
188 public class MainClass {
189     public static void main(String[] args) {
190         String configFile = "classpath:applicationContext.xml";
191         AbstractApplicationContext ctx = new
GenericXmlApplicationContext(configFile);
192         MyCalculator myCalculator = ctx.getBean("myCalculator",
MyCalculator.class);
193
194         myCalculator.add();
195         myCalculator.sub();
196         myCalculator.multi();
197         myCalculator.div();
198
199         ctx.close();
200     }
201 }

```

9. Result

```

205 BeforeSpring과 같음.
206 Called addAction()
207 10 + 2 = 12
208 Called subAction()
209 10 - 2 = 8
210 Called multiAction()
211 10 x 2 = 20
212 Called divAction()
213 10 / 2 = 5

```

Task 3. 간단한 DI Project

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

275 5. Printer interface의 child class 작성하기

```
276 1)com.example > right-click > New > Class
277   -Name : StringPrinter
278   -Interfaces : com.example.Printer
279   -Finish
280
281 2)StringPrinter.java
282   package com.example;
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
297 3)com.example > right-click > New > Class
298   -Name : ConsolePrinter
299   -Interface : com.example.Printer
300   -Finish
301
```

```
302 4)ConsolePrinter.java
303   package com.example;
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
314 6. Java Project를 Spring Project로 변환
```

```
315 1)DIDemo Project > right-click > Configure > Convert to Maven Project
316   -Project : /DIDemo
317   -Group Id : DIDemo
318   -Artifact Id : DIDemo
319   -version : 0.0.1-SNAPSHOT
320   -Packaging : jar
321   -Finish
322
```

```
323 2)DIDemo Project > right-click > Spring > Add Spring Project Nature
324
```

```
325 3)pom.xml file에 Spring Context Dependency 추가하기
326   <version>0.0.1-SNAPSHOT</version>
327   <dependencies>
328       <dependency>
329           <groupId>org.springframework</groupId>
330           <artifactId>spring-context</artifactId>
331           <version>5.3.10</version>
```

```
332         </dependency>
333     </dependencies>
334
335     4)pom.xml > right-click > Run As > Maven install
336         [INFO] BUILD SUCCESS 확인
337
338
339     7. config folder 생성
340         1)StartSpring project > right-click > New > Source Folder
341             -Folder name : config
342             -Finish
343
344
345     8. Bean Configuration XML 작성
346         1)config > right-click > New > Other > Spring > Spring Bean Configuration File
347             > Next
348         2)File name : beans.xml
349         3)Finish
350
351         <?xml version="1.0" encoding="UTF-8"?>
352         <beans xmlns="http://www.springframework.org/schema/beans"
353             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
354             xsi:schemaLocation="http://www.springframework.org/schema/beans
355                 http://www.springframework.org/schema/beans/spring-beans.xsd">
356
357             <bean id="hello" class="com.example.Hello">
358                 <property name="name" value="Spring" />
359                 <property name="printer" ref="printer" />
360             </bean>
361             <bean id="printer" class="com.example.StringPrinter" />
362             <bean id="consolePrinter" class="com.example.ConsolePrinter" />
363
364         </beans>
365
366     9. Beans Graph 사용하기
367         1)Window menu > Show View > Other > Spring > Spring Explorer > Open
368         2)Spring Explorer
369             -DIDemo > Beans > beans.xml > right-click > Open Beans Graph
370
371
372     10. DI Test class 작성
373         1)src/com.example > right-click > New > Package
374             -Name : com.example.test
375             -Finish
376
377         2)/src/com.example.test > New > Class
378             -Name : HelloBeanTest.java
379
380             package com.example.test;
381
382             import org.springframework.context.ApplicationContext;
383             import org.springframework.context.support.GenericXmlApplicationContext;
384
385             import com.example.Hello;
386             import com.example.Printer;
```

```

386
387 public class HelloBeanTest {
388     public static void main(String [] args){
389         //1. IoC Container 생성
390         ApplicationContext context =
391             new GenericXmlApplicationContext("classpath:beans.xml");
392
393         //2. Hello Beans 가져오기
394         Hello hello = (Hello)context.getBean("hello");
395         System.out.println(hello.sayHello());
396         hello.print();
397
398         //3. SpringPrinter 가져오기
399         Printer printer = (Printer)context.getBean("printer");
400         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 }

```

11. Result

```

Hello Spring
Hello Spring
true

```

Task 4. JUnit을 사용한 DI test class 작성하기

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

1)pom.xml에 아래 코드 붙여넣기

```

<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.13.2</version>
    <scope>test</scope>
</dependency>

```

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

[INFO] BUILD SUCCESS 확인

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

-Name : HelloBeanJUnitTest.java

```
package com.example.test;
```

```
import org.junit.Before;
```

```
import org.junit.Test;
```

```
import org.springframework.context.ApplicationContext;
```

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

```
import com.example.Hello;
```



```

import com.example.Printer;

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

public class HelloBeanJUnitTest {
    private ApplicationContext context;

    @Before
    public void init(){
        //항상 먼저 ApplicationContext를 생성해야 하기 때문에
        //1. IoC Container 생성
        context = new GenericXmlApplicationContext("classpath:beans.xml");
    }

    @Test
    public void test1(){
        //2. Hello Beans 가져오기
        Hello hello = (Hello)context.getBean("hello");
        assertEquals("Hello Spring", hello.sayHello());
        hello.print();

        //3. SpringPrinter 가져오기
        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);
        assertSame(hello, hello2);
    }
}

```

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

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

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

-Add JUnit 4 library to the build path > OK

3. right-click > Run As > JUnit Test

1)결과 -> JUnit View에 초록색 bar

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

```
import org.junit.Ignore;
```

```
...
```

```
@Test @Ignore
```

```
public void test1(){
```

```
...
```

3)right-click > Run As > JUnit Test

-JUnit Test 목록에서 test1()는 실행되지 않는다.

```

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
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.assertSame;
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;
535         import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
536
537         import com.example.Hello;
538         import com.example.Printer;
539
540         @RunWith(SpringJUnit4ClassRunner.class)
541         //JUnit 4.x에서 사용
542         @ContextConfiguration(locations="classpath:beans.xml")
543         public class HelloBeanJUnitSpringTest {
544             @Autowired
545             ApplicationContext context;
546
547             @Test
548             public void test1(){
549                 Hello hello = (Hello)context.getBean("hello");
550                 assertEquals("Hello Spring", hello.sayHello());
551                 hello.print();
552
553                 Printer printer = (Printer)context.getBean("printer");

```

```

554         assertEquals("Hello Spring", printer.toString());
555     }
556
557     @Test
558     public void test2(){
559         Hello hello = (Hello)context.getBean("hello");
560
561         Hello hello2 = context.getBean("hello", Hello.class);
562         assertEquals(hello, hello2);
563     }
564 }

```

565
566 -right-click > Run As > JUnit Test
567 -결과 -> JUnit View에 초록색 bar

- 568
- 569 4)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
- 570 -해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
 - 571 -spring-test-5.3.10.jar 선택 후 [Remove] 로 삭제
 - 572 -Classpath 선택
 - 573 -[Add External JARs...] Click
 - 574 -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
 - 575 -[Order and Export] tab에서 spring-test-5.3.10.jar 선택 후 [Up] button을 클릭
 - 576 -해당 DIDemo/src 바로 아래까지 올리고 [Apply and Close] Click

577
578
579
580 -----
581 Task 6. Java Annotation을 이용하여 setter를 이용한 의존주입하기 실습

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

583 1)Project name : DIDemo1

584 2)JRE

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

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

587 4)Next

588 5)Finish

589
590
591 2. src > right-click > New > Package

592 1)Package name : com.example

593 2)Finish

594
595
596 3. POJO class 작성

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

598 2)Name : Hello

599
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 3)com.example > right-click > New > Interface

625 4)Name : Printer

```
626
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
```

```
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 }

```

7. DI Test class 작성

- 1)src > right-click > New > Package
- 2)Package Name : com.example.test
- 3)Finish
- 4)com.example.test > right-click > New > Class
- 5)Name : HelloBeanTest

```

741     package com.example.test;
742
743     import org.springframework.context.ApplicationContext;
744     import
745     org.springframework.context.annotation.AnnotationConfigApplicationContext;
746
747     import com.example.Hello;
748     import com.example.Printer;
749     import com.example.config.AppCtx;
750
751     public class HelloBeanTest {
752         public static void main(String[] args) {
753             // 1. IoC Container 생성
754             ApplicationContext ctx = new
755             AnnotationConfigApplicationContext(AppCtx.class);
756
757             // 2. Hello Beans 가져오기
758             Hello hello = (Hello)ctx.getBean("hello");
759             System.out.println(hello.sayHello());
760             hello.print();
761
762             // 3. StringPrinter 가져오기
763             Printer printer = (Printer) ctx.getBean("printer");
764             System.out.println(printer.toString());
765             Hello hello2 = ctx.getBean("hello", Hello.class);
766             hello2.print();
767             System.out.println(hello == hello2);
768         }
769     }

```

8. Test

- 1)/src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application
- 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
797     package com.example;
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

```

839
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     }

```

4. Java Project를 Spring Project로 변환

1)SpringDemo Project > right-click > Configure > 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>
941
942
943 7. MainClass 생성하기
944     1)com.example.MainClass.java
945         package com.example;
946
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         }
964
965

```

966 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.
973
974

```

```

975 -----

```

```

976 [추가 lab] : PropertyEditor 실습

```

977 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
984
985

```

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

```

987     1)Package name : com.example
988
989

```

990 3. POJO class 작성

```

991     1)com.example > right-click > New > Class
992     2)Class Name : SimpleBean
993
994

```

```

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 > Configure > 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

```
1106 2)Source Tab > Add Folder
1107 3)PropertyEditorDemo 선택 확인
1108 4)Create New Folder > Folder name : resources > Finish > OK
1109 5)PropertyEditorDemo/resources(new) 확인
1110 6)Apply and Close
1111
1112
1113 6. Bean Configuration XML 작성
1114 1)PropertyEditorDemo/resources > right-click > New > Other > Spring >
Spring Bean Configuration File
1115 2)File name : applicationContext.xml
1116 3)Finish
1117
1118 <!-- 실제 bean에 대한 정의 -->
1119 <bean id="simpleBean" class="com.example.SimpleBean">
1120     <!-- property type에 맞게 알아서 PropertyEditor가 동작한다. -->
1121     <property name="bytes">
1122         <value>Hello, World</value>
1123     </property>
1124     <property name="cls">
1125         <value>java.lang.String</value>
1126     </property>
1127     <property name="trueOrFalse">
1128         <value>true</value>
1129     </property>
1130     <property name="stringList">
1131         <util:list>
1132             <value>String member 1</value>
1133             <value>String member 2</value>
1134         </util:list>
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
1163 7. MainClass 생성하기
1164     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
```

1177 8. Java Application 실행

1178
1179 -----

1180 Task 8. 생성자 이용하여 의존 주입하기 실습

1181 1. In Package Explorer > right-click > New > Java Project

1182 1)Project name : DIDemo2

1183 2)JRE

1184 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

1185 3)Uncheck [Create module-info.java file]

1186 4)Next

1187 5)Finish

1188

1189

1190 2. src > right-click > New > Package

1191 1)Package name : com.example

1192 2)Finish

1193

1194

1195 3. POJO class 작성

1196 1)com.example > right-click > New > Class

1197 2)Class Name : Hello

1198 package com.example;

1199

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)Interfaces : 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
1302 Configuration File
1303 -File name : beans.xml > Finish
1304
1305 <?xml version="1.0" encoding="UTF-8"?>
1306 <beans xmlns="http://www.springframework.org/schema/beans"
1307 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">
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

```

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

```

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

```

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

```

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     ...
```

12. Test

1) /src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application

Hello Spring

Hello Spring

true

Task 9. Java Annotation을 이용한 생성자 이용하여 의존 주입하기 실습

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

1) Project Name : SpringDemo1

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 생성

1) com.example.Student.java

```
package com.example;
```

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

2) com.example.StudentInfo.java

```
package com.example;
```

```
public class StudentInfo {
    private Student student;
}
```

4. Java Project를 Spring Project로 변환

1) SpringDemo1 Project > right-click > Configure > Convert to Maven Project

-Project : /SpringDemo1

-Group Id : SpringDemo1

-Artifact Id : SpringDemo1

-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
1480 6. Student.java와 StudentInfo.java 수정
1481 1)Student.java
1482
1483     package com.example;
1484
1485     import lombok.Getter;
1486     import lombok.Setter;
1487     import lombok.ToString;
1488     import lombok.AllArgsConstructor;
1489
1490     @Getter
1491     @Setter
```

```
1492 @ToString
1493 @AllArgsConstructor
1494 public class Student {
1495     private String name;
1496     private int age;
1497     private int grade;
1498     private int classNum;
1499 }
```

1501 2)StudentInfo.java

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

1525 7. 환경설정을 위해 config package 생성

- 1526 1)com.example package > right-click > New > Package
- 1527 2)Name : com.example.config
- 1528 3)Finish

1531 8. ApplicationContext.java 생성

- 1532 1)com.example.config > right-click > New > Class
- 1533 2)Name : ApplicationCtx
- 1534 3)Finish

```
1535
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 }

```

9. com.example.MainClass.java

```

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

```

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 -----

```

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
1670     [INFO] BUILD SUCCESS 확인
1671
1672
1673 5. Lombok library 추가
1674     1)https://mvnrepository.com/에서 'lombok'으로 검색
1675     2)'Project Lombok' click
1676     3)1.18.20 click
1677     4)dependency copy해서 pom.xml에 붙여넣기
1678
1679     <dependencies>
1680         <dependency>
1681             <groupId>org.springframework</groupId>
1682             <artifactId>spring-context</artifactId>
1683             <version>5.3.10</version>
1684         </dependency>
1685         <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
1686         <dependency>
1687             <groupId>org.projectlombok</groupId>
1688             <artifactId>lombok</artifactId>
1689             <version>1.18.20</version>
1690             <scope>provided</scope>
1691         </dependency>
1692     </dependencies>
1693
1694 5)pom.xml > right-click > Run As > Maven install
1695     [INFO] BUILD SUCCESS 확인
1696
1697
1698 6. SpringDemo2/resources folder 생성
1699     1)SpringDemo2 project > right-click > new > Source Folder
1700     2)Folder Name : resources
1701     3)Finish
1702
1703
1704 7. Bean Configuration XML 작성
1705     1)resources Folder > right-click > New > Spring Bean Configuration File
1706     2)File name : applicationContext.xml > Finish
1707     3)resources Folder > right-click > New > Spring Bean Configuration File
1708     4)File name : applicationContext2.xml > Finish
1709
1710
1711 8. Student.java, StudentInfo.java 그리고 Product.java에 lombok Annotation 붙이기
1712     1)Student.java
1713

```

```
1714 package com.example;
1715
1716 import java.util.ArrayList;
1717
1718 import lombok.AllArgsConstructor;
1719 import lombok.Data;
1720 import lombok.NonNull;
1721 import lombok.RequiredArgsConstructor;
1722
1723 @Data
1724 @RequiredArgsConstructor
1725 @AllArgsConstructor
1726 public class Student {
1727     private @NonNull String name;
1728     private @NonNull int age;
1729     private @NonNull ArrayList<String> hobbies;
1730     private double height;
1731     private double weight;
1732 }
```

1734 2)StudentInfo.java

```
1735
1736 package com.example;
1737
1738 import lombok.Setter;
1739 import lombok.Getter;
1740
1741 @Setter
1742 @Getter
1743 public class StudentInfo {
1744     private Student student;
1745 }
```

1747 3)Product.java

```
1748
1749 package com.example;
1750
1751 import lombok.AllArgsConstructor;
1752 import lombok.NoArgsConstructor;
1753 import lombok.NonNull;
1754 import lombok.RequiredArgsConstructor;
1755 import lombok.Setter;
1756 import lombok.ToString;
1757
1758 @NoArgsConstructor
1759 @AllArgsConstructor
1760 @RequiredArgsConstructor
1761 @Setter
1762 @ToString
1763 public class Product {
1764     private @NonNull String pName;
1765     private @NonNull int pPrice;
1766     private String maker;
1767     private String color;
1768 }
1769
```



```

1770
1771 9. applicationContext.xml
1772     <?xml version="1.0" encoding="UTF-8"?>
1773     <beans xmlns="http://www.springframework.org/schema/beans"
1774           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1775           xsi:schemaLocation="http://www.springframework.org/schema/beans
1776                               http://www.springframework.org/schema/beans/spring-beans.xsd">
1777
1778         <bean id="student1" class="com.example.Student">
1779             <constructor-arg value="백두산" />
1780             <constructor-arg value="25" />
1781             <constructor-arg>
1782                 <list>
1783                     <value>독서</value>
1784                     <value>영화감상</value>
1785                     <value>요리</value>
1786                 </list>
1787             </constructor-arg>
1788             <property name="height" value="165" />
1789             <property name="weight">
1790                 <value>45</value>
1791             </property>
1792         </bean>
1793
1794         <bean id="studentInfo1" class="com.example.StudentInfo">
1795             <property name="student">
1796                 <ref bean="student1" />
1797             </property>
1798         </bean>
1799     </beans>
1800
1801 10. applicationContext2.xml
1802     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
1811         <bean id="student3" class="com.example.Student">
1812             <constructor-arg value="한라산" />
1813             <constructor-arg value="50" />
1814             <constructor-arg>
1815                 <list>
1816                     <value>노래부르기</value>
1817                     <value>게임</value>
1818                 </list>
1819             </constructor-arg>
1820             <property name="height" value="175" />
1821             <property name="weight">
1822                 <value>75</value>
1823             </property>
1824         </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>
1830
1831 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
1842             GenericXmlApplicationContext(configFile, configFile1);
1843             Student student1 = context.getBean("student1", Student.class);
1844             System.out.println(student1);
1845
1846             StudentInfo studentInfo = context.getBean("studentInfo1",
1847             StudentInfo.class);
1848             Student student2 = studentInfo.getStudent();
1849             System.out.println(student2);
1850             if(student1.equals(student2)) System.out.println("Equals");
1851             else System.out.println("Different");
1852
1853             Student student3 = context.getBean("student3", Student.class);
1854             System.out.println(student3);
1855
1856             if(student1.equals(student3)) System.out.println("Equals");
1857             else System.out.println("Different");
1858
1859             Product product = context.getBean("product", Product.class);
1860             System.out.println(product);
1861             context.close();
1862         }
1863     }
1864
1865 12. Java Application 실행
1866     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리],
1867     height=165.0,weight=45.0]
1868     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리],
1869     height=165.0,weight=45.0]
1870     Equals
1871     Student [name=한라산, age=50, hobbies=[노래부르기, 게임],
1872     height=175.0,weight=75.0]
1873     Different
1874     Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]
1875 -----

```

1874 Task 11. Java Annotation을 이용하여 두 개 이상의 설정 파일로 DI 설정하기
1875 1. In Package Explorer > right-click > New > Java Project
1876 1)Project Name : SpringDemo3
1877 2)JRE
1878 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
1879 3)Uncheck [Create module-info.java file]
1880 4)Next
1881 5)Finish
1882
1883
1884 2)src > right-click > New > Package
1885 1)Package name : com.example
1886 2)Finish
1887
1888
1889 3. POJO Class 생성
1890 1)com.example.Student.java
1891
1892 package com.example;
1893
1894 import java.util.ArrayList;
1895
1896 public class Student {
1897 private String name;
1898 private int age;
1899 private ArrayList<String> hobbies;
1900 private double height;
1901 private double weight;
1902 }
1903
1904 2)com.example.StudentInfo.java
1905
1906 package com.example;
1907 public class StudentInfo {
1908 private Student student;
1909 }
1910
1911 3)com.example.Product.java
1912
1913 package com.example;
1914 public class Product {
1915 private String pName;
1916 private int pPrice;
1917 private String maker;
1918 private String color;
1919 }
1920
1921 4. Java Project를 Spring Project로 변환
1922 1)SpringDemo3 Project > right-click > Configure > Convert to Maven Project
1923 -Project : /SpringDemo3
1924 -Group Id : SpringDemo3
1925 -Artifact Id : SpringDemo3
1926 -version : 0.0.1-SNAPSHOT
1927 -Packaging : jar
1928 -Finish
1929

2)SpringDemo3 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. 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
@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.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 @NonNull String maker;
```

```
2042     private String color;
2043 }
2044
2045
```

2046 9. AppConfig1.java

```
2047 package com.example.config;
2048
2049 import java.util.Arrays;
2050 import java.util.List;
2051
2052 import org.springframework.context.annotation.Bean;
2053 import org.springframework.context.annotation.Configuration;
2054
2055 import com.example.Student;
2056 import com.example.StudentInfo;
2057
2058 @Configuration
2059 public class AppConfig1 {
2060     @Bean
2061     public Student student1() {
2062         List<String> list = Arrays.asList("독서", "영화감상", "요리");
2063         Student student1 = new Student("백두산", 25, list);
2064         student1.setHeight(165);
2065         student1.setWeight(45);
2066         return student1;
2067     }
2068
2069     @Bean
2070     public StudentInfo studentInfo1() {
2071         StudentInfo studentInfo1 = new StudentInfo();
2072         studentInfo1.setStudent(this.student1());
2073         return studentInfo1;
2074     }
2075 }
2076
2077
```

2078 10. AppConfig2.java

```
2079 package com.example.config;
2080
2081 import java.util.Arrays;
2082 import java.util.List;
2083
2084 import org.springframework.context.annotation.Bean;
2085 import org.springframework.context.annotation.Configuration;
2086
2087 import com.example.Product;
2088 import com.example.Student;
2089
2090 @Configuration
2091 public class AppConfig2 {
2092     @Bean
2093     public Student student3() {
2094         List<String> list = Arrays.asList("노래부르기", "게임");
2095         Student student3 = new Student("한라산", 50, list);
2096         student3.setHeight(175);
2097         student3.setWeight(75);

```

```

2098         return student3;
2099     }
2100
2101     @Bean
2102     public Product product() {
2103         Product product = new Product("Computer", 2000000, "Samsung");
2104         product.setColor("Yellow");
2105         return product;
2106     }
2107 }
2108
2109
2110 11. com.example.MainClass
2111     package com.example;
2112
2113     import org.springframework.context.ApplicationContext;
2114     import
2115     org.springframework.context.annotation.AnnotationConfigApplicationContext;
2116
2117     import com.example.config.AppConfig1;
2118     import com.example.config.AppConfig2;
2119
2120     public class MainClass {
2121         public static void main(String[] args) {
2122             ApplicationContext context = new
2123             AnnotationConfigApplicationContext(AppConfig1.class, AppConfig2.class);
2124             Student student1 = context.getBean("student1", Student.class);
2125             System.out.println(student1);
2126
2127             StudentInfo studentInfo = context.getBean("studentInfo1",
2128             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);
2135             System.out.println(student3);
2136
2137             if(student1.equals(student3)) System.out.println("Equals");
2138             else System.out.println("Different");
2139
2140             Product product = context.getBean("product", Product.class);
2141             System.out.println(product);
2142         }
2143     }
2144
2145 12. Java Application 실행
2146     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리],
2147     height=165.0,weight=45.0]
2148     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리],
2149     height=165.0,weight=45.0]
2150     Equals
2151     Student [name=한라산, age=50, hobbies=[노래부르기, 게임],
2152     height=175.0,weight=75.0]

```

2148 Different
2149 Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]
2150
2151
2152

2153 -----

2154 Task 12. Java Annotation과 XML 을 이용한 DI 설정 방법 : XML file에 Java file을 포함시켜
사용하는 방법

2155 1. In Package Explorer > right-click > New > Java Project

2156 1)Project Name : SpringDemo4

2157 2)JRE

2158 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

2159 3)Uncheck [Create module-info.java file]

2160 4)Next

2161 5)Finish

2162

2163

2164 2. src > right-click > New > Package

2165 1)Package name : com.example

2166

2167

2168 3. POJO 생성

2169 1)com.example.Student.java

2170 package com.example;

2171

2172 import java.util.ArrayList;

2173

2174 public class Student {

2175 private String name;

2176 private int age;

2177 private ArrayList<String> hobbies;

2178 private double height;

2179 private double weight;

2180 }

2181

2182

2183 4. Java Project를 Spring Project로 변환

2184 1)SpringDemo4 Project > right-click > Configure > Convert to Maven Project

2185 -Project : /SpringDemo4

2186 -Group Id : SpringDemo4

2187 -Artifact Id : SpringDemo4

2188 -version : 0.0.1-SNAPSHOT

2189 -Packaging : jar

2190 -Finish

2191

2192 2)SpringDemo4 Project > right-click > Spring > Add Spring Project Nature

2193

2194 3)pom.xml file에 Spring Context Dependency 추가하기

2195 <version>0.0.1-SNAPSHOT</version>

2196 <dependencies>

2197 <dependency>

2198 <groupId>org.springframework</groupId>

2199 <artifactId>spring-context</artifactId>

2200 <version>5.3.10</version>

2201 </dependency>

2202 </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 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;
}
```

7. com.example.ApplicationConfig.java

```
package com.example;
```

```

2259
2260 import java.util.Arrays;
2261 import java.util.List;
2262
2263 import org.springframework.context.annotation.Bean;
2264 import org.springframework.context.annotation.Configuration;
2265
2266 @Configuration
2267 public class ApplicationConfig {
2268     @Bean
2269     public Student student1(){
2270         List<String> list = Arrays.asList("독서", "영화감상", "요리");
2271
2272         Student student1 = new Student("백두산", 25, list);
2273         student1.setHeight(165);
2274         student1.setWeight(45);
2275
2276         return student1;
2277     }
2278 }

```

8. SpringDemo4/resources folder 생성

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

9. Bean Configuration XML 작성

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

10. /resources/applicationContext.xml

```

2296 <?xml version="1.0" encoding="UTF-8"?>
2297 <beans xmlns="http://www.springframework.org/schema/beans"
2298     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2299     xmlns:context="http://www.springframework.org/schema/context"
2300     xsi:schemaLocation="http://www.springframework.org/schema/beans
2301         http://www.springframework.org/schema/beans/spring-beans.xsd">
2302
2303     <bean
2304         class="org.springframework.context.annotation.ConfigurationClassPostProcess
2305         or" />
2306
2307     <bean class="com.example.ApplicationConfig" />
2308     <bean id="student3" class="com.example.Student">
2309         <constructor-arg value="북한산" />
2310         <constructor-arg value="50" />
2311         <constructor-arg>
2312             <list>
2313                 <value>노래부르기</value>
2314                 <value>게임</value>
2315             </list>

```

```

2312         </constructor-arg>
2313         <property name="height" value="175" />
2314         <property name="weight">
2315             <value>75</value>
2316         </property>
2317     </bean>
2318 </beans>

```

2321 11. com.example.MainClass.java

```

2322     package com.example;
2323
2324     import org.springframework.context.support.AbstractApplicationContext;
2325     import org.springframework.context.support.GenericXmlApplicationContext;
2326
2327     public class MainClass {
2328         public static void main(String[] args) {
2329             String configFile = "classpath:applicationContext.xml";
2330             AbstractApplicationContext context = new
                GenericXmlApplicationContext(configFile);
2331             Student student1 = context.getBean("student1", Student.class);
2332             System.out.println(student1);
2333
2334             Student student3 = context.getBean("student3", Student.class);
2335             System.out.println(student3);
2336         }
2337     }

```

2340 12. Java Application 실행

```

2341     Student [name=백두산, age=25, hobbies=[독서, 영화감상, 요리],
        height=165.0,weight=45.0]
2342     Student [name=북한산, age=50, hobbies=[노래부르기, 게임],
        height=175.0,weight=75.0]

```

2345 13. JUnit을 사용한 DI test class 작성하기

```

2346     1)com.example > right-click > New > JUnit Test Case
2347     2)Select [New JUnit 4 test]
2348     3)Name : HelloBeanJUnitTest
2349     4)Finish
2350     5)[New JUnit Test Case] 창에서 Select [Perform the follwing action:] > Add JUnit 4
        library to the build path
2351     6)OK

```

2354 14. JUnit을 사용한 Test

```

2355     1)src/com.example > New > Class
2356         -Name : HelloBeanJUnitTest.java
2357
2358     package com.example;
2359
2360     import static org.junit.Assert.assertEquals;
2361     import static org.junit.Assert.assertSame;
2362
2363     import org.junit.Before;

```

```

2364 import org.junit.Test;
2365 import org.springframework.context.ApplicationContext;
2366 import org.springframework.context.support.GenericXmlApplicationContext;
2367
2368 public class HelloBeanJUnitTest {
2369     private ApplicationContext context;
2370
2371     @Before
2372     public void init(){
2373         context = new
                GenericXmlApplicationContext("classpath:applicationContext.xml");
2374     }
2375
2376     @Test
2377     public void test1(){
2378         Student student1 = (Student)context.getBean("student1");
2379         assertEquals("백두산", student1.getName());
2380         System.out.println(student1);
2381     }
2382
2383     @Test
2384     public void test2(){
2385         Student student3 = context.getBean("student3", Student.class);
2386         System.out.println(student3);
2387
2388         Student student4 = (Student)context.getBean("student3");
2389         assertEquals(student3, student4);
2390     }
2391 }
2392
2393 2)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test
2394     -JUnit 창에 Green Bar
2395         Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,
                weight=45.0)
2396         Student(name=북한산, age=50, hobbies=[노래부르기, 게임], height=175.0,
                weight=75.0)
2397
2398
2399 15. Spring TestContext Framework을 이용한 Test
2400 1)Spring-Test library 설치
2401     -http://mvnrepository.com에서 'spring test'로 검색
2402     -검색 결과 목록에서 'Spring TestContext Framework' Click
2403     -version 목록에서 5.3.10 Click
2404
2405 2)dependency 복사해서 pom.xml에 붙여넣기
2406     <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
2407     <dependency>
2408         <groupId>org.springframework</groupId>
2409         <artifactId>spring-test</artifactId>
2410         <version>5.3.10</version>
2411         <scope>test</scope>
2412     </dependency>
2413
2414 3)pom.xml > right-click > Maven Install
2415     [INFO] BUILD SUCCESS
2416

```

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
    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)에
서 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

2471 Task 13. Java Annotation과 XML 을 이용한 DI 설정 방법 : Java file에 XML file을 포함시켜
사용하는 방법

2472 1. In Package Explorer > right-click > New > Java Projectn

2473 1)Project Name : SpringDemo5

2474 2)JRE

2475 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]

2476 3)Uncheck [Create module-info.java file]

2477 4)Next

2478 5)Finish

2479

2480

2481 2. src > right-click > New > Package

2482 1)Package name : com.example

2483 2)Finish

2484

2485

2486 3. com.example.Student.java

2487 package com.example;

2488

2489 import java.util.List;

2490

2491 public class Student {

2492 private String name;

2493 private int age;

2494 private List<String> hobbies;

2495 private double height;

2496 private double weight;

2497 }

2498

2499

2500 4. Java Project를 Spring Project로 변환

2501 1)SpringDemo5 Project > right-click > Configure > Convert to Maven Project

2502 -Project : /SpringDemo5

2503 -Group Id : SpringDemo5

2504 -Artifact Id : SpringDemo5

2505 -version : 0.0.1-SNAPSHOT

2506 -Packaging : jar

2507 -Finish

2508

2509 2)SpringDemo5 Project > right-click > Spring > Add Spring Project Nature

2510

2511 3)pom.xml file에 Spring Context Dependency 추가하기

2512 <version>0.0.1-SNAPSHOT</version>

2513 <dependencies>

2514 <dependency>

2515 <groupId>org.springframework</groupId>

2516 <artifactId>spring-context</artifactId>

2517 <version>5.3.10</version>

2518 </dependency>

2519 </dependencies>

2520

2521 4)pom.xml > right-click > Run As > Maven install

2522 [INFO] BUILD SUCCESS 확인

2523

2524

2525 5. Lombok library 추가

```
2526 1)https://mvnrepository.com/에서 'lombok'으로 검색
2527 2)'Project Lombok' click
2528 3)1.18.20 click
2529 4)dependency copy해서 pom.xml에 붙여넣기
2530
2531 <dependencies>
2532   <dependency>
2533     <groupId>org.springframework</groupId>
2534     <artifactId>spring-context</artifactId>
2535     <version>5.3.10</version>
2536   </dependency>
2537   <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
2538   <dependency>
2539     <groupId>org.projectlombok</groupId>
2540     <artifactId>lombok</artifactId>
2541     <version>1.18.20</version>
2542     <scope>provided</scope>
2543   </dependency>
2544 </dependencies>
```

```
2545
2546 5)pom.xml > right-click > Run As > Maven install
2547 [INFO] BUILD SUCCESS 확인
2548
2549
```

2550 6. Student.java lombok Annotation 붙이기

```
2551 1)Student.java
2552
2553 package com.example;
2554
2555 import java.util.List;
2556
2557 import lombok.AllArgsConstructor;
2558 import lombok.Data;
2559 import lombok.NonNull;
2560 import lombok.RequiredArgsConstructor;
2561
2562 @Data
2563 @RequiredArgsConstructor
2564 @AllArgsConstructor
2565 public class Student {
2566     private @NonNull String name;
2567     private @NonNull int age;
2568     private @NonNull List<String> hobbies;
2569     private double height;
2570     private double weight;
2571 }
2572
2573
```

2574 7. SpringDemo5/resources folder 생성

```
2575 1)SpringDemo5 project > right-click > Build Path > Configure Build Path
2576 2)Source Tab > Add Folder
2577 3)SpringDemo5 선택 확인
2578 4>Create New Folder > Folder name : resources > Finish > OK
2579 5)SpringDemo5/resources(new) 확인
2580 6)Apply and Close
2581
```

```

2582
2583 8. Bean Configuration XML 작성
2584 1)SpringDemo5/resources > right-click > New > Spring Bean Configuration File
2585 2)File name : applicationContext.xml > Finish
2586
2587
2588 9. /resources/applicationContext.xml
2589 <?xml version="1.0" encoding="UTF-8"?>
2590 <beans xmlns="http://www.springframework.org/schema/beans"
2591     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2592     xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
2593
2594     <bean id="student3" class="com.example.Student">
2595         <constructor-arg value="지리산" />
2596         <constructor-arg value="30" />
2597         <constructor-arg>
2598             <list>
2599                 <value>등산</value>
2600                 <value>게임</value>
2601                 <value>독서</value>
2602             </list>
2603         </constructor-arg>
2604         <property name="height" value="165" />
2605         <property name="weight">
2606             <value>49</value>
2607         </property>
2608     </bean>
2609 </beans>
2610
2611
2612 10. com.example.ApplicationConfig.java
2613 package com.example;
2614
2615 import java.util.Arrays;
2616 import java.util.List;
2617
2618 import org.springframework.context.annotation.Bean;
2619 import org.springframework.context.annotation.Configuration;
2620 import org.springframework.context.annotation.ImportResource;
2621
2622 @Configuration
2623 @ImportResource("classpath:ApplicationContext.xml")
2624 public class ApplicationConfig {
2625     @Bean
2626     public Student student1(){
2627         List<String> hobbies = Arrays.asList("독서", "영화감상", "요리");
2628
2629         Student student = new Student("백두산", 25, hobbies);
2630         student.setHeight(165);
2631         student.setWeight(45);
2632
2633         return student;
2634     }
2635 }
2636

```



```

2637
2638 11. com.example.MainClass.java
2639     package com.example;
2640
2641     import
2642     org.springframework.context.annotation.AnnotationConfigApplicationContext;
2643
2644     public class MainClass {
2645         public static void main(String[] args) {
2646             AnnotationConfigApplicationContext context = new
2647             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 12. Java Application 실행
2659     Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,
2660     weight=45.0)
2661     Student(name=지리산, age=30, hobbies=[등산, 게임, 독서], height=165.0,
2662     weight=49.0)
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
2670     library to the build path
2671     6)OK
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

```

2688 -만일 ERROR 발생하면 다음과 같이 조치한다.
2689 -SpringDemo5 > right-click > Maven > Update Project
2690 -SpringDemo5가 check되어 있음을 확인하고 OK
2691 -다시 pom.xml > right-click > Maven Install
2692 [INFO] BUILD SUCCESS
2693
2694

15. JUnit 5를 사용한 Test

1)com.example.HelloBeanJUnitTest.java

```
package com.example;

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

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

class HelloBeanJUnitTest {
    private ApplicationContext context;

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

    @Test
    public void test1(){
        Student student1 = (Student)context.getBean("student1");
        assertEquals("백두산", student1.getName());
        System.out.println(student1);
    }

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

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

-JUnit 창에 Green Bar

```
Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,
weight=45.0)
Student(name=홍지민, age=30, hobbies=[등산, 게임, 독서], height=165.0,
weight=49.0)
```

2740 Task 14. Lab
2741 1. In Package Explorer > right-click > New > Java Project
2742 1)Project name : DIDemo3
2743 2)JRE
2744 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
2745 3)Uncheck [Create module-info.java file]
2746 4)Next
2747 5)Finish
2748
2749
2750 2. src > right-click > New > Package
2751 1)Package name : com.example
2752 2)Finish
2753
2754
2755 3. POJO class 작성
2756 1)com.example > right-click > New > Class
2757 2)Class Name : Hello
2758
2759 package com.example;
2760
2761 public class Hello{
2762 private String name;
2763 private Printer printer;
2764
2765 public String sayHello(){
2766 return "Hello " + name;
2767 }
2768
2769 public void print(){
2770 this.printer.print(sayHello());
2771 }
2772 }
2773
2774 3)com.example > right-click > New > Interface
2775 4)interface name : Printer
2776
2777 package com.example;
2778
2779 public interface Printer{
2780 void print(String message);
2781 }
2782
2783 5)com.example > right-click > New > Class
2784 6)Class Name : StringPrinter
2785 7)Interfaces : com.example.Printer
2786
2787 package com.example;
2788
2789 public class StringPrinter implements Printer{
2790 private StringBuffer buffer = new StringBuffer();
2791
2792 @Override
2793 public void print(String message){
2794 this.buffer.append(message);
2795 }

```
2796
2797     public String toString(){
2798         return this.buffer.toString();
2799     }
2800 }
2801
```

2802 8)com.example > right-click > New > Class

2803 9)Class Name : ConsolePrinter

2804 10)Interfaces : com.example.Printer

```
2805
2806     package com.example;
2807
2808     public class ConsolePrinter implements Printer{
2809
2810         @Override
2811         public void print(String message){
2812             System.out.println(message);
2813         }
2814     }
2815
```

```
2816
2817 4. Java Project를 Spring Project로 변환
2818     1)DIDemo3 Project > right-click > Configure > Convert to Maven Project
2819         -Project : /DIDemo3
2820         -Group Id : DIDemo3
2821         -Artifact Id : DIDemo3
2822         -version : 0.0.1-SNAPSHOT
2823         -Packaging : jar
2824         -Finish
2825
```

```
2826     2)DIDemo3 Project > right-click > Spring > Add Spring Project Nature
2827
```

```
2828     3)pom.xml file에 Spring Context Dependency 추가하기
2829         <version>0.0.1-SNAPSHOT</version>
2830         <dependencies>
2831             <dependency>
2832                 <groupId>org.springframework</groupId>
2833                 <artifactId>spring-context</artifactId>
2834                 <version>5.3.10</version>
2835             </dependency>
2836         </dependencies>
2837
```

```
2838     4)pom.xml > right-click > Run As > Maven install
2839         [INFO] BUILD SUCCESS 확인
2840
```

```
2841
2842 5. Lombok library 추가
2843     1)https://mvnrepository.com/에서 'lombok'으로 검색
2844     2)'Project Lombok' click
2845     3)1.18.20 click
2846     4)dependency copy해서 pom.xml에 붙여넣기
2847
```

```
2848         <dependencies>
2849             <dependency>
2850                 <groupId>org.springframework</groupId>
2851                 <artifactId>spring-context</artifactId>
```

```
2852         <version>5.3.10</version>
2853     </dependency>
2854     <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
2855     <dependency>
2856         <groupId>org.projectlombok</groupId>
2857         <artifactId>lombok</artifactId>
2858         <version>1.18.20</version>
2859         <scope>provided</scope>
2860     </dependency>
2861 </dependencies>
```

2863 5)pom.xml > right-click > Run As > Maven install

2864 [INFO] BUILD SUCCESS 확인

2867 6. Hello.java에 lombok Annotation으로 수정하기

```
2868 package com.example;
2869
2870 import lombok.NoArgsConstructor;
2871 import lombok.Setter;
2872
2873 @Setter
2874 @NoArgsConstructor
2875 public class Hello {
2876     private String name;
2877     private Printer printer;
2878
2879     public String sayHello(){
2880         return "Hello " + name;
2881     }
2882
2883     public void print(){
2884         this.printer.print(sayHello());
2885     }
2886 }
2887 }
```

2890 7. src/config folder 생성

2891 1)/src > right-click > New > Folder

2892 2)Folder name : config

2895 8. Bean Configuration XML 작성

2896 1)/src/config > right-click > New > Other > Spring > Spring Bean Configuration File

2897 2)File name : beans.xml > Finish

```
2898
2899 <?xml version="1.0" encoding="UTF-8"?>
2900 <beans xmlns="http://www.springframework.org/schema/beans"
2901     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2902     xsi:schemaLocation="http://www.springframework.org/schema/beans
2903         http://www.springframework.org/schema/beans/spring-beans.xsd">
2904     <bean id="hello" class="com.example.Hello">
2905         <property name="name" value="Spring" />
```

```

2906         <property name="printer" ref="printer" />
2907     </bean>
2908     <bean id="printer" class="com.example.StringPrinter" />
2909     <bean id="consolePrinter" class="com.example.ConsolePrinter" />
2910 </beans>

```

9. DI Test class 작성

```

2914 1)/src > right-click > New > Package
2915 2)Name : com.example.test
2916 3)Finish
2917 4)/src/com.example.test > right-click > New > Class
2918 5)Name : HelloBeanTest
2919
2920     package com.example.test;
2921
2922     import org.springframework.context.ApplicationContext;
2923     import org.springframework.context.support.GenericXmlApplicationContext;
2924
2925     import com.example.Hello;
2926     import com.example.Printer;
2927
2928     public class HelloBeanTest {
2929         public static void main(String [] args){
2930             ApplicationContext context = new
                GenericXmlApplicationContext("config/beans.xml");
2931
2932             Hello hello = (Hello)context.getBean("hello");
2933             System.out.println(hello.sayHello());
2934             hello.print();
2935
2936             Printer printer = (Printer)context.getBean("printer");
2937             System.out.println(printer.toString());
2938
2939             Hello hello2 = context.getBean("hello", Hello.class);
2940             hello2.print();
2941
2942             System.out.println(hello == hello2); //Singleton Pattern
2943         }
2944     }

```

6)Java Application 실행

```

2947     Hello Spring
2948     Hello Spring
2949     true

```

10. JUnit 5 Library 설치

```

2953 1)JUnit 5 설치
2954     -http://mvnrepository.com에서 'junit'로 검색
2955     -검색 결과 목록에서 'JUnit Jupiter API' Click
2956     -version 목록에서 5.8.1 click
2957
2958 2)dependency 복사해서 pom.xml에 붙여넣기
2959     <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
2960     <dependency>

```

```
2961     <groupId>org.junit.jupiter</groupId>
2962     <artifactId>junit-jupiter-api</artifactId>
2963     <version>5.8.1</version>
2964     <scope>test</scope>
2965 </dependency>
```

2967 3)pom.xml > right-click > Maven Install

2968 [INFO] BUILD SUCCESS

2969 -만일 ERROR 발생하면 다음과 같이 조치한다.

2970 -SpringDemo5 > right-click > Maven > Update Project

2971 -SpringDemo5가 check되어 있음을 확인하고 OK

2972 -다시 pom.xml > right-click > Maven Install

2973 [INFO] BUILD SUCCESS

2976 11. JUnit 5를 사용한 Test

2977 1)com.example.test > right-click > New > Class

2978 2)Name : HelloBeanJUnitTest

2980 package com.example.test;

2982 import static org.junit.jupiter.api.Assertions.assertEquals;

2983 import static org.junit.jupiter.api.Assertions.assertSame;

2985 import org.junit.jupiter.api.BeforeEach;

2986 import org.junit.jupiter.api.Test;

2987 import org.springframework.context.ApplicationContext;

2988 import org.springframework.context.support.GenericXmlApplicationContext;

2990 import com.example.Hello;

2992 public class HelloBeanJUnitTest {

2993 private ApplicationContext context;

2995 @BeforeEach

2996 public void init() {

2997 this.context = new GenericXmlApplicationContext("config/beans.xml");

2998 }

3000 @Test

3001 public void test1(){

3002 Hello hello = (Hello)context.getBean("hello");

3003 assertEquals("Hello Spring", hello.sayHello());

3004 hello.print();

3005 }

3007 @Test

3008 public void test2(){

3009 Hello hello = (Hello)context.getBean("hello");

3010 Hello hello2 = context.getBean("hello", Hello.class);

3011 assertSame(hello, hello2);

3012 }

3013 }

3016 3)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면

3017 -해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
3018 -Classpath 선택
3019 -[Add External JARs...] Click
3020 -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
3021 -[Order and Export] tab에서 junit-jupiter-api-5.8.1.jar 선택 후 [Up] button을 클릭
3022 -해당 Project/src 바로 아래까지 올리고 [Apply and Close] Click
3023

3024 4)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test
3025 -JUnit 창에 Green Bar
3026

3028 12. Spring TestContext Framework

3029 1)Spring-Test library 설치

3030 2)pom.xml 수정

```
3031  
3032 <dependency>  
3033 <groupId>org.springframework</groupId>  
3034 <artifactId>spring-test</artifactId>  
3035 <version>5.3.10</version>  
3036 <scope>test</scope>  
3037 </dependency>
```

3038
3039 3)pom.xml > right-click > Maven Install

3040 -만일 Error 발생시 DIDemo3 > right-click > Maven > Update Project... > Ok
3041 -다시 Maven Install 실행
3042

3043 4)Spring-Test를 사용할 DI test class-HelloBeanJUnitSpringTest.java 작성하기

3044 -/src/com.example.test > New > Class

3045 -Name : HelloBeanJUnitSpringTest

3046 -Finish
3047

```
3048 package com.example.test;
```

```
3049  
3050 import static org.junit.jupiter.api.Assertions.assertEquals;
```

```
3051 import static org.junit.jupiter.api.Assertions.assertSame;
```

```
3052  
3053 import org.junit.jupiter.api.Test;
```

```
3054 import org.junit.jupiter.api.extension.ExtendWith;
```

```
3055 import org.springframework.beans.factory.annotation.Autowired;
```

```
3056 import org.springframework.context.ApplicationContext;
```

```
3057 import org.springframework.test.context.ContextConfiguration;
```

```
3058 import org.springframework.test.context.junit.jupiter.SpringExtension;
```

```
3059  
3060 import com.example.Hello;
```

```
3061  
3062 @ExtendWith(SpringExtension.class)
```

```
3063 //JUnit 5.x에서 사용
```

```
3064 @ContextConfiguration(locations="classpath:config/beans.xml")
```

```
3065 public class HelloBeanJUnitSpringTest {
```

```
3066     @Autowired
```

```
3067     ApplicationContext context;
```

```
3068  
3069     @Test
```

```
3070     public void test1(){
```



```

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

```

```

3125 //import javax.annotation.Resource;
3126
3127 import javax.inject.Inject;
3128 import javax.inject.Named;
3129
3130 import lombok.NoArgsConstructor;
3131 import lombok.Setter;
3132
3133 @Setter
3134 @NoArgsConstructor
3135 @Component
3136 public class Hello {
3137     @Value("Spring")
3138     private String name;
3139
3140     //@Autowired(required = true)
3141     //@Qualifier("stringPrinter")
3142     //@Resource(name = "stringPrinter")
3143     /*
3144     @Resource annotation 사용하려면 mvnrepository.com에서 Javax Annotation
3145     API 검색해서
3146     <dependency>
3147         <groupId>javax.annotation</groupId>
3148         <artifactId>javax.annotation-api</artifactId>
3149         <version>1.3.2</version>
3150     </dependency>
3151
3152     @Inject annotation 사용하려면 mvnrepository.com에서 Javax Inject API 검색해서
3153     <dependency>
3154         <groupId>javax.inject</groupId>
3155         <artifactId>javax.Inject</artifactId>
3156         <version>1</version>
3157     </dependency>
3158     */
3159     @Inject
3160     @Named("stringPrinter")
3161     private Printer printer;
3162
3163     public String sayHello(){
3164         return "Hello " + name;
3165     }
3166
3167     public void print(){
3168         this.printer.print(sayHello());
3169     }
3170 }
3171

```

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

17. 새로운 설정 file 생성

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

2)File name : annos.xml > Finish

3)Namespace tab > context Check

```

3179 <?xml version="1.0" encoding="UTF-8"?>
3180 <beans xmlns="http://www.springframework.org/schema/beans"
3181       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3182       xmlns:context="http://www.springframework.org/schema/context"
3183       xsi:schemaLocation="http://www.springframework.org/schema/beans
3184       http://www.springframework.org/schema/beans/spring-beans.xsd
3185       http://www.springframework.org/schema/context
3186       http://www.springframework.org/schema/context/spring-context-4.3.xsd">
3187     <context:component-scan base-package="com.example" />
3188 </beans>

```

18. /src/com.example.test/HelloBeanJUnitSpringTest.java 수정하기

```

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

```

1)right-click > Run As > Junit Test

2)결과 -> Junit View에 초록색 bar

3233 Task 15. Lab with JUnit 5 Jupiter
3234 1. In Package Explorer > right-click > New > Java Project
3235 1)Project Name : DIDemo4
3236 2)JRE
3237 -Select [Use default JRE 'jdk-11.0.12' and workspace compiler preferences]
3238 3)Uncheck [Create module-info.java file]
3239 4)Next
3240 5)Finish
3241
3242
3243 2. src > right-click > New > Package
3244 1)Package name : com.example
3245 2)Finish
3246
3247
3248 3. com.example.Student.java, com.example.StudentInfo.java
3249 1)Student.java
3250 package com.example;
3251
3252 import java.util.List;
3253
3254 public class Student {
3255 private String name;
3256 private int age;
3257 private List<String> hobbies;
3258 private double height;
3259 private double weight;
3260 }
3261
3262 2)StudentInfo.java
3263 package com.example;
3264
3265 public class StudentInfo {
3266 private Student student;
3267 }
3268
3269
3270 4. Java Project를 Spring Project로 변환
3271 1)DIDemo4 Project > right-click > Configure > Convert to Maven Project
3272 -Project : /DIDemo4
3273 -Group Id : DIDemo4
3274 -Artifact Id : DIDemo4
3275 -version : 0.0.1-SNAPSHOT
3276 -Packaging : jar
3277 -Finish
3278
3279 2)DIDemo4 Project > right-click > Spring > Add Spring Project Nature
3280
3281 3)pom.xml file에 Spring Context Dependency 추가하기
3282 <version>0.0.1-SNAPSHOT</version>
3283 <dependencies>
3284 <dependency>
3285 <groupId>org.springframework</groupId>
3286 <artifactId>spring-context</artifactId>
3287 <version>5.3.10</version>
3288 </dependency>

```

3289     </dependencies>
3290
3291 4)pom.xml > right-click > Run As > Maven install
3292     [INFO] BUILD SUCCESS 확인
3293
3294
3295 5. Lombok library 추가
3296 1)https://mvnrepository.com/에서 'lombok'으로 검색
3297 2)'Project Lombok' click
3298 3)1.18.20 click
3299 4)dependency copy해서 pom.xml에 붙여넣기
3300
3301     <dependencies>
3302         <dependency>
3303             <groupId>org.springframework</groupId>
3304             <artifactId>spring-context</artifactId>
3305             <version>5.3.10</version>
3306         </dependency>
3307         <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
3308         <dependency>
3309             <groupId>org.projectlombok</groupId>
3310             <artifactId>lombok</artifactId>
3311             <version>1.18.20</version>
3312             <scope>provided</scope>
3313         </dependency>
3314     </dependencies>
3315
3316 5)pom.xml > right-click > Run As > Maven install
3317     [INFO] BUILD SUCCESS 확인
3318
3319
3320 6. Student.java, StudentInfo.java lombok Annotation 붙이기
3321 1)Student.java
3322
3323     package com.example;
3324
3325     import java.util.List;
3326
3327     import lombok.AllArgsConstructor;
3328     import lombok.Data;
3329     import lombok.NonNull;
3330     import lombok.RequiredArgsConstructor;
3331
3332     @Data
3333     @RequiredArgsConstructor
3334     @AllArgsConstructor
3335     public class Student {
3336         private @NonNull String name;
3337         private @NonNull int age;
3338         private @NonNull List<String> hobbies;
3339         private double height;
3340         private double weight;
3341     }
3342
3343 2)StudentInfo.java
3344

```

```

3345 package com.example;
3346
3347 import lombok.AllArgsConstructor;
3348 import lombok.Setter;
3349
3350 @Setter
3351 @AllArgsConstructor
3352 public class StudentInfo {
3353     private Student student;
3354
3355     public void printInfo(){
3356         if(this.student != null){
3357             System.out.println("Name : " + this.student.getName());
3358             System.out.println("Age : " + this.student.getAge());
3359             System.out.println("Hobbies");
3360             this.student.getHobbys().forEach(hobby ->
3361                 System.out.println(hobby));
3362             System.out.println("Height : " + this.student.getHeight());
3363             System.out.println("Weight : " + this.student.getWeight());
3364         }
3365     }
3366 }

```

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

```

3379 package com.example.config;
3380
3381 import java.util.Arrays;
3382 import java.util.List;
3383
3384 import org.springframework.context.annotation.Bean;
3385 import org.springframework.context.annotation.Configuration;
3386
3387 import com.example.Student;
3388 import com.example.StudentInfo;
3389
3390 @Configuration
3391 public class ApplicationConfig {
3392     @Bean
3393     public Student student1() {
3394         List<String> list = Arrays.asList("독서", "영화감상", "요리");
3395         Student student1 = new Student("백두산", 25, list);
3396         student1.setHeight(165);
3397         student1.setWeight(45);
3398         return student1;
3399     }

```

```

3400
3401     @Bean
3402     public StudentInfo studentInfo() {
3403         return new StudentInfo(this.student1());
3404     }
3405 }
3406
3407
3408 9. com.example.MainClass.java
3409
3410 package com.example;
3411
3412 import
3413 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",
3425             StudentInfo.class);
3426         studentInfo.setStudent(student1);
3427         studentInfo.printInfo();
3428
3429         context.close();
3430     }
3431 }
3432
3433 10. Java Application 실행
3434 Student(name=백두산, age=25, hobbies=[독서, 영화감상, 요리], height=165.0,
3435 weight=45.0)
3436 Name : 백두산
3437 Age : 25
3438 Hobbies
3439 독서
3440 영화감상
3441 요리
3442 Height : 165.0
3443 Weight : 45.0
3444
3445 11. Student.java 수정
3446
3447 package com.example;
3448
3449 import java.util.List;
3450
3451 import org.springframework.beans.factory.annotation.Value;
3452 import org.springframework.stereotype.Component;

```

```

3452 import lombok.Getter;
3453 import lombok.Setter;
3454
3455 @Component
3456 @Setter
3457 @Getter
3458 public class Student {
3459     @Value("백두산")
3460     private String name;
3461     @Value("25")
3462     private int age;
3463     @Value("등산, 게임, 독서")
3464     private List<String> hobbies;
3465     @Value("162.5")
3466     private double height;
3467     @Value("49.2")
3468     private double weight;
3469 }
3470
3471

```

3472 12. StudentInfo.java 수정

```

3473
3474 package com.example;
3475
3476 import org.springframework.beans.factory.annotation.Autowired;
3477 import org.springframework.stereotype.Component;
3478
3479 import lombok.NoArgsConstructor;
3480 import lombok.Setter;
3481
3482 @NoArgsConstructor
3483 @Component
3484 public class StudentInfo {
3485     @Setter(onMethod_ = @Autowired)
3486     private Student student;
3487
3488     public void printInfo(){
3489         if(this.student != null){
3490             System.out.println("Name : " + this.student.getName());
3491             System.out.println("Age : " + this.student.getAge());
3492             System.out.println("Hobbies");
3493             this.student.getHobbys().forEach(hobby -> System.out.println(hobby));
3494             System.out.println("Height : " + this.student.getHeight());
3495             System.out.println("Weight : " + this.student.getWeight());
3496         }else {
3497             System.out.println("Null");
3498         }
3499     }
3500 }
3501
3502

```

3503 13. ApplicationConfig.java 수정

```

3504
3505 package com.example.config;
3506
3507 import org.springframework.context.annotation.Bean;

```



```
3508 import org.springframework.context.annotation.ComponentScan;
3509 import org.springframework.context.annotation.Configuration;
3510
3511 import com.example.StudentInfo;
3512
3513 @Configuration
3514 @ComponentScan(basePackages = {"com.example"})
3515 public class ApplicationConfig {
3516     @Bean
3517     public StudentInfo studentInfo() {
3518         return new StudentInfo();
3519     }
3520 }
3521
3522
3523 14. MainClass.java 수정
3524
3525 package com.example;
3526
3527 import
3528 org.springframework.context.annotation.AnnotationConfigApplicationContext;
3529
3530 import com.example.config.ApplicationConfig;
3531
3532 public class MainClass {
3533     public static void main(String[] args) {
3534         AnnotationConfigApplicationContext context = new
3535             AnnotationConfigApplicationContext(ApplicationConfig.class);
3536         StudentInfo info = context.getBean("studentInfo", StudentInfo.class);
3537         info.printInfo();
3538         context.close();
3539     }
3540 }
3541
3542 15. MainClass 실행
3543
3544 Name : 백두산
3545 Age : 25
3546 Hobbies
3547 등산, 게임, 독서
3548 Height : 162.5
3549 Weight : 49.2
```