HOL : Spring DI

Task 1. Non-DI Java Project
1. Project 유형 : Java Project
2. Project Name : BeforeSpring
3. Package Name : com.example

4. Calculator Class
com.example.Calculator.java

```
package com.example;

public class Calculator {
  public void addAction(int a, int b){
    System.out.println("Called addAction()");
    System.out.printf("%d + %d = %d\n", a, b, (a + b));
  }
  public void subAction(int a, int b){
    System.out.println("Called subAction()");
    System.out.printf("%d - %d = %d\n", a, b, (a - b));
  }
  public void multiAction(int a, int b){
    System.out.println("Called multiAction()");
    System.out.printf("%d x %d = %d\n", a, b, (a * b));
  }
  public void divAction(int a, int b){
    System.out.println("Called divAction()");
    System.out.printf("%d / %d = %d\n", a, b, (a / b));
  }
}
```

5. MyCalculator Class
com.example.MyCalculator.java

```
package com.example;

public class MyCalculator {
  private Calculator calculator;
  private int firstNum;
  private int secondNum;

  public void setFirstNum(int firstNum) {
    this.firstNum = firstNum;
  }
  public void setSecondNum(int secondNum) {
    this.secondNum = secondNum;
  }
  public void setCalculator(Calculator calculator){
    this.calculator = calculator;
  }

  public void add(){
    this.calculator.addAction(firstNum, secondNum);
  }
  public void sub(){
    this.calculator.subAction(firstNum, secondNum);
  }
  public void multi(){
```

```
        this.calculator.multiAction(firstNum, secondNum);
      }
      public void div(){
        this.calculator.divAction(firstNum, secondNum);
      }
    }


6. MainClass Class
   com.example.MainClass
   package com.example;

   public class MainClass {
     public static void main(String[] args) {
       MyCalculator myCalculator = new MyCalculator();
       myCalculator.setCalculator(new Calculator());

       myCalculator.setFirstNum(10);
       myCalculator.setSecondNum(2);

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


7. Result
   Called addAction()
   10 + 2 = 12
   Called subAction()
   10 - 2 = 8
   Called multiAction()
   10 x 2 = 20
   Called divAction()
   10 / 2 = 5



-----------------------------------------
Task 2. DI Demo in Spring
1. New > Java Project
   1)Project Name : StartSpring
   2)JRE : Use default JRE 'jdk-13.0.2' and workspace compiler preferences
   3)Next
   4)Uncheck [Create module-info.java file]
   5)Finish


2. Create package to src : com.example

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

4. Create class : com.example.MainClass.java
   package com.example;

   public class MainClass {
```

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

    }
}


5. Java Project를 Spring Project로 변환
  1)StartSpring Project > right-click > Configure > Convert to Maven Project
    -Project : /StartSpring
    -Group Id : StartSpring
    -Artifact Id : StartSpring
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish
    -Package Explorer에서 보이는 Project icon에 Maven의 'M'자가 보임.

  2)StartSpring Project > right-click > Spring > Add Spring Project Nature
    -Package Explorer에서 보이는 Project icon에 'M'자와 Spring의 'S'가 보임.

  3)pom.xml file에 Spring Context Dependency 추가하기
    -https://mvnrepository.com에서 'spring context'로 검색
    -[Spring Context] click
    -현재 Spring 5.x의 현재 version인 5.2.5.RELEASE click
    -Copy하여 pom.xml에 paste

   <version>0.0.1-SNAPSHOT</version>
    <dependencies>   <--- dependencies element 추가
      <dependency>   <---여기에 paste
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.2.5.RELEASE</version>
      </dependency>
    </dependencies>

  4)pom.xml Save

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


6. config folder 생성
  1)StartSpring project > right-click > New > Source Folder
    -Folder name : config
    -Finish


7. Bean Configuration XML 작성
  1)config > right-click > New > Other > Spring > Spring Bean Configuration File > Next
  2)Name : applicationContext.xml > Finish
    <?xml version="1.0" encoding="UTF-8"?>
    <beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
      http://www.springframework.org/schema/beans/spring-beans.xsd">

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

      <bean id="myCalculator" class="com.example.MyCalculator">
```

```
        <property name="calculator">
          <ref bean="calculator" />
        </property>
        <property name="firstNum" value="10" />
        <property name="secondNum" value="2" />
      </bean>
    </beans>


8. MainClass.java
  package com.javasoft;

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

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

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

      ctx.close();
    }
  }


9. Result
  BeforeSpring과 같음.
  Called addAction()
  10 + 2 = 12
  Called subAction()
  10 - 2 = 8
  Called multiAction()
  10 x 2 = 20
  Called divAction()
  10 / 2 = 5



----------------------------------------------------
Task 3. 간단한 DI Project
1. In Package Explorer > right-click > New > Java Project
  1)Project name : DIDemo
  2)JRE : Use default JRE 'jdk-13.0.2' and workspace compiler preferences
  3)Next
  4)Uncheck [Create module-info.java file]
  5)Finish


2. src > right-click > New > Package
  1)Package name : com.example
  2)Finish

```

3. Interface 작성
  1)com.example > right-click > New > Interface
  2)Name : Printer

  3)Printer.java
```
package com.example;

public interface Printer{
  void print(String message);
}
```

4. POJO class 작성
  1)com.example > right-click > New > Class
  2)Name : Hello
  3)Finish
  4)Hello.java
```
package com.example;

public class Hello{
  private String name;
  private Printer printer;

  public Hello(){}

  public void setName(String name){
    this.name = name;
  }

  public void setPrinter(Printer printer){
    this.printer = printer;
  }

  public String sayHello(){
    return "Hello " + name;
  }

  public void print(){
    this.printer.print(sayHello());
  }
}
```

5. Printer interface의 child class 작성하기
  1)com.example > right-click > New > Class
    -Name : StringPrinter
    -Interfaces : com.example.Printer
    -Finish

  2)StringPrinter.java
```
package com.example;

public class StringPrinter implements Printer{
  private StringBuffer buffer = new StringBuffer();

  @Override
  public void print(String message){
```

```
290         this.buffer.append(message);
291       }
292
293       public String toString(){
294         return this.buffer.toString();
295       }
296     }
297
298   3)com.example > right-click > New > Class
299     -Name : ConsolePrinter
300     -Interface : com.example.Printer
301     -Finish
302
303   4)ConsolePrinter.java
304     package com.example;
305
306     public class ConsolePrinter implements Printer{
307
308       @Override
309       public void print(String message){
310         System.out.println(message);
311       }
312     }
313
314
315 6. Java Project를 Spring Project로 변환
316   1)DIDemo Project > right-click > Configure > Convert to Maven Project
317     -Project : /DIDemo
318     -Group Id : DIDemo
319     -Artifact Id : DIDemo
320     -version : 0.0.1-SNAPSHOT
321     -Packaging : jar
322     -Finish
323
324   2)DIDemo Project > right-click > Spring > Add Spring Project Nature
325
326   3)pom.xml file에 Spring Context Dependency 추가하기
327     <version>0.0.1-SNAPSHOT</version>
328     <dependencies>
329       <dependency>
330         <groupId>org.springframework</groupId>
331         <artifactId>spring-context</artifactId>
332         <version>5.2.5.RELEASE</version>
333       </dependency>
334     </dependencies>
335
336   4)pom.xml > right-click > Run As > Maven install
337     [INFO] BUILD SUCCESS 확인
338
339
340 7. config folder 생성
341   1)StartSpring project > right-click > New > Source Folder
342     -Folder name : config
343     -Finish
344
345
346 8. Bean Configuration XML 작성
347   1)config > right-click > New > Other > Spring > Spring Bean Configuration File > 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
         http://www.springframework.org/schema/beans/spring-beans.xsd">
355
356        <bean id="hello" class="com.example.Hello">
357          <property name="name" value="Spring" />
358          <property name="printer" ref="printer" />
359        </bean>
360        <bean id="printer" class="com.example.StringPrinter" />
361        <bean id="consolePrinter" class="com.example.ConsolePrinter" />
362
363      </beans>
364
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;
387
388      public class HelloBeanTest {
389        public static void main(String [] args){
390          //1. IoC Container 생성
391          ApplicationContext context =
392               new GenericXmlApplicationContext("classpath:beans.xml");
393
394          //2. Hello Beans 가져오기
395          Hello hello = (Hello)context.getBean("hello");
396          System.out.println(hello.sayHello());
397          hello.print();
398
399          //3. SpringPrinter 가져오기
400          Printer printer = (Printer)context.getBean("printer");
401          System.out.println(printer.toString());
402
403          Hello hello2 = context.getBean("hello", Hello.class);
404          hello2.print();
```

```
405
406           System.out.println(hello == hello2);  //Singleton Pattern
407         }
408       }
409
410
411   11. Result
412     Hello Spring
413     Hello Spring
414     true
415
416
417
418   ------------------------------
419   Task 4. JUnit을 사용한 DI test class 작성하기
420   1. JUnit을 사용한 DI test class(HelloBeanJunitTest.java) 작성
421     1)pom.xml에 아래 코드 붙여넣기
422       <dependency>
423         <groupId>junit</groupId>
424         <artifactId>junit</artifactId>
425         <version>4.12</version>
426         <scope>test</scope>
427       </dependency>
428
429     2)pom.xml > right-click > Run As > Maven install
430       [INFO] BUILD SUCCESS 확인
431
432     3)src/com.example.test > New > Class
433       -Name : HelloBeanJUnitTest.java
434
435       package com.example.test;
436
437       import org.junit.Before;
438       import org.junit.Test;
439       import org.springframework.context.ApplicationContext;
440       import org.springframework.context.support.GenericXmlApplicationContext;
441
442       import com.example.Hello;
443       import com.example.Printer;
444
445       import static org.junit.Assert.assertEquals;
446       import static org.junit.Assert.assertSame;
447
448       public class HelloBeanJUnitTest {
449         ApplicationContext context;
450
451         @Before
452         public void init(){
453           //항상 먼저 ApplicationContext를 생성해야 하기 때문에
454           //1. IoC Container 생성
455           context = new GenericXmlApplicationContext("classpath:beans.xml");
456         }
457
458         @Test
459         public void test1(){
460           //2. Hello Beans 가져오기
461           Hello hello = (Hello)context.getBean("hello");
462           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 import org.junit.Ignore;
    ...
    @Test @Ignore
    public void test1(){
    ...

  3)right-click > Run As > Junit Test
    -JUnit Test 목록에서 test1()는 실행되지 않는다.



-----------------------------------------
Task 5. Spring TestContext Framework
1. Spring-Test library 설치
  1)http://mvnrepository.com에서 'spring test'로 검색
  2)검색 결과 목록에서 'Spring TestContext Framework' Click
  3)version 목록에서 5.2.5.RELEASE Click


2. dependency 복사해서 pom.xml에 붙여넣기
  <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-test</artifactId>
    <version>5.2.5.RELEASE</version>
    <scope>test</scope>
  </dependency>


3. pom.xml > right-click > Maven Install
  [INFO] BUILD SUCCESS
```

4. Spring-Test를 사용할 DI test class-HelloBeanJunitSpringTest.java 작성하기
  1)src/com.example.test > New > Class
  2)Name : HelloBeanJunitSpringTest

```
package com.example.test;

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;

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

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

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

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

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

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

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

  4)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
  -해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
  -spring-test-5.2.5.RELEASE.jar 선택 후 [Remove] 로 삭제
  -Classpath 선택
  -[Add External JARs...] Click
  -Local M2 Repository(e.g C:\Users\bluee\.m2\repository\org\springframework\spring-test\5.2.5.RELEASE)에서 직접 jar(spring-test-5.2.5.RELEASE.jar)를 선택할 것
  -[Order and Export] tab에서 spring-test-5.2.5.RELEASE.jar 선택 후 [Up] button을 클릭

```
577      -해당 DIDemo/src 바로 아래까지 올리고 [Apply and Close] Click
578
579
580
581 ----------------------------------------
582 Task 6. Java Annotation을 이용하여 setter를 이용한 의존주입하기 실습
583 1. In Package Explorer > right-click > New > Java Project
584   1)Project name : DIDemo1
585   2)JRE
586     -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
587   3)Next
588   4)Uncheck [Create module-info.java file]
589   5)Finish
590
591
592 2. src > right-click > New > Package
593   1)Package name : com.example
594   2)Finish
595
596
597 3. POJO class 작성
598   1)com.example > right-click > New > Class
599   2)Name : Hello
600
601     package com.example;
602
603     public class Hello {
604       private String name;
605       private Printer printer;
606
607       public Hello(){}
608
609       public void setName(String name){
610         this.name = name;
611       }
612
613       public void setPrinter(Printer printer){
614         this.printer = printer;
615       }
616
617       public String sayHello(){
618         return "Hello " + name;
619       }
620
621       public void print(){
622         this.printer.print(sayHello());
623       }
624     }
625
626   3)com.example > right-click > New > Interface
627   4)Name : Printer
628
629     package com.example;
630
631     public interface Printer{
632       void print(String message);
633     }
634
```

```
   5)com.example > right-click > New > Class
   6)Name : StringPrinter
   7)Interfaces : com.example.Printer

     package com.example;

     public class StringPrinter implements Printer{
       private StringBuffer buffer = new StringBuffer();

       @Override
       public void print(String message){
         this.buffer.append(message);
       }

       public String toString(){
         return this.buffer.toString();
       }
     }

   8)com.example > right-click > New > Class
   9)Name : ConsolePrinter
   10)Interfaces : com.example.Printer

     package com.example;

     public class ConsolePrinter implements Printer{

       @Override
       public void print(String message){
         System.out.println(message);
       }
     }


4. Java Project를 Spring Project로 변환
   1)DIDemo1 Project > right-click > Configure > Convert to Maven Project
     -Project : /DIDemo1
     -Group Id : DIDemo1
     -Artifact Id : DIDemo1
     -version : 0.0.1-SNAPSHOT
     -Packaging : jar
     -Finish

   2)DIDemo1 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>5.2.5.RELEASE</version>
       </dependency>
     </dependencies>

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

```

5. config package 생성
  1)com.example > right-click > New > Package > com.example.config
  2)Finish

6. AppCtx Class 생성
  1)com.example.config > right-click > New > Class
  2)Name : AppCtx.java

```
package com.example.config;

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

import com.example.ConsolePrinter;
import com.example.Hello;
import com.example.StringPrinter;

@Configuration
public class AppCtx {

  @Bean
  public Hello hello() {
    Hello hello = new Hello();
    hello.setName("Spring");
    hello.setPrinter(this.printer());
    return hello;
  }

  @Bean
  public StringPrinter printer() {
    return new StringPrinter();
  }

  @Bean
  public ConsolePrinter consolePrinter() {
    return new ConsolePrinter();
  }
}
```

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

```
package com.example.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.example.Hello;
import com.example.Printer;
import com.example.config.AppCtx;
```

```
    public class HelloBeanTest {
      public static void main(String[] args) {
        // 1. IoC Container 생성
        ApplicationContext ctx = new AnnotationConfigApplicationContext(AppCtx.class);

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

        // 3. SpringPrinter 가져오기
        Printer printer = (Printer) ctx.getBean("printer");
        System.out.println(printer.toString());
        Hello hello2 = ctx.getBean("hello", Hello.class);
        hello2.print();
        System.out.println(hello == hello2);
      }
    }


8. Test
  1)/src/com.example.test/HelloBeanTest.java > right-click > Run As > Java Application
    Hello Spring
    Hello Spring
    true



---------------------------------------------
Task 7. setter를 이용한 의존주입하기 실습
1. In Package Explorer > right-click > New > Java Project
  1)Project Name : SpringDemo
  2)JRE
    -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
  3)Next
  4)Uncheck [Create module-info.java file]
  5)Finish


2. src > right-click > New > Package
  1)Package name : com.example


3. POJO class 작성
  1)com.example > right-click > New > Class
  2)Class Name : BmiCalculator

    package com.example;

    public class BmiCalculator {
      private double lowWeight;
      private double normal;
      private double overWeight;
      private double obesity;

      public void setLowWeight(double lowWeight) {
        this.lowWeight = lowWeight;
      }
```

```
809
810         public void setNormal(double normal) {
811           this.normal = normal;
812         }
813
814         public void setOverWeight(double overWeight) {
815           this.overWeight = overWeight;
816         }
817
818         public void setObesity(double obesity) {
819           this.obesity = obesity;
820         }
821         public void bmiCalcu(double weight, double height){
822           double h = height * 0.01;
823           double result = weight / (h * h);
824
825           System.out.println("BMI 지수 : " + (int)result);
826
827           if(result > obesity)
828             System.out.println("비만입니다.");
829           else if(result > overWeight)
830             System.out.println("과체중입니다.");
831           else if(result > normal)
832             System.out.println("정상입니다.");
833           else
834             System.out.println("저체중입니다.");
835         }
836       }
837
838     3)com.example > right-click > New > Class
839     4)Class Name : MyInfo.java
840
841       package com.example;
842
843       import java.util.ArrayList;
844
845       public class MyInfo {
846         private String name;
847         private double height;
848         private double weight;
849         private ArrayList<String> hobby;
850         private BmiCalculator bmiCalculator;
851
852         public void setBmiCalculator(BmiCalculator bmiCalculator) {
853           this.bmiCalculator = bmiCalculator;
854         }
855         public void setName(String name) {
856           this.name = name;
857         }
858         public void setHeight(double height) {
859           this.height = height;
860         }
861         public void setWeight(double weight) {
862           this.weight = weight;
863         }
864         public void setHobby(ArrayList<String> hobby) {
865           this.hobby = hobby;
866         }
```

```
867         public void getInfo(){
868           System.out.println("Name : " + this.name);
869           System.out.println("Height : " + this.height);
870           System.out.println("Weight : " + this.weight);
871           System.out.println("Hobby : " + this.hobby);
872           this.bmiCalcu();
873         }
874         public void bmiCalcu(){
875           this.bmiCalculator.bmiCalcu(this.weight, this.height);
876         }
877       }
878
879
880   4. Java Project를 Spring Project로 변환
881     1)SpringDemo Project > right-click > Configue > Convert to Maven Project
882       -Project : /SpringDemo
883       -Group Id : SpringDemo
884       -Artifact Id : SpringDemo
885       -version : 0.0.1-SNAPSHOT
886       -Packaging : jar
887       -Finish
888
889     2)SpringDemo Project > right-click > Spring > Add Spring Project Nature
890
891     3)pom.xml file에 Spring Context Dependency 추가하기
892       <version>0.0.1-SNAPSHOT</version>
893       <dependencies>
894         <dependency>
895           <groupId>org.springframework</groupId>
896           <artifactId>spring-context</artifactId>
897           <version>5.2.5.RELEASE</version>
898         </dependency>
899       </dependencies>
900
901     4)pom.xml > right-click > Run As > Maven install
902       [INFO] BUILD SUCCESS 확인
903
904
905   5. SpringDemo/resources folder 생성
906     1)SpringDemo project > right-click > Build Path > Configure Build Path
907     2)Source Tab > Add Folder
908     3)SpringDemo 선택 확인
909     4)Create New Folder > Folder name : resources > Finish > OK
910     5)SpringDemo/resources(new) 확인
911     6)Apply and Close
912
913
914   6. Bean Configuration XML 작성
915     1)SpringDemo/resources > right-click > New > Other > Spring > Spring Bean Configuration
      File
916     2)File name : applicationContext.xml
917     3)Finish
918
919       <bean id="bmiCalculator" class="com.example.BmiCalculator">
920         <property name="lowWeight" value="18.5" />
921         <property name="normal" value="23" />
922         <property name="overWeight" value="25" />
923         <property name="obesity">
```

```
        <value>30</value>
      </property>
    </bean>
    <bean id="myInfo" class="com.example.MyInfo">
      <property name="name" value="한지민" />
      <property name="height" value="170.5" />
      <property name="weight" value="67" />
      <property name="hobby">
        <list>
          <value>수영</value>
          <value>요리</value>
          <value>독서</value>
        </list>
      </property>
      <property name="bmiCalculator">
        <ref bean="bmiCalculator" />
      </property>
    </bean>


7. MainClass 생성하기
  1)com.example.MainClass.java
    package com.example;

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

    public class MainClass {
      public static void main(String[] args) {
        String configFile = "classpath:applicationContext.xml";

        //Spring Container 생성
        AbstractApplicationContext context = new GenericXmlApplicationContext(configFile);

        //Spring Container 에서 객체를 가져옴
        MyInfo myInfo = context.getBean("myInfo", MyInfo.class);

        myInfo.getInfo();
        context.close();
      }
    }


8. Java Application 실행
  Name : 한지민
  Height : 170.5
  Weight : 67.0
  Hobby : [수영, 요리, 독서]
  BMI 지수 : 23
  정상입니다.



----------------------------------------
Task 8. 생성자 이용하여 의존 주입하기 실습
1. In Package Explorer > right-click > New > Java Project
  1)Project name : DIDemo2
  2)JRE
```

```
982       -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
983     3)Next
984     4)Uncheck [Create module-info.java file]
985     5)Finish
986
987
988   2. src > right-click > New > Package
989     1)Package name : com.example
990     2)Finish
991
992
993   3. POJO class 작성
994     1)com.example > right-click > New > Class
995     2)Class Name : Hello
996       package com.example;
997
998       public class Hello{
999         private String name;
1000        private Printer printer;
1001
1002        public Hello(){}
1003
1004        public void setName(String name){
1005          this.name = name;
1006        }
1007
1008        public void setPrinter(Printer printer){
1009          this.printer = printer;
1010        }
1011
1012        public String sayHello(){
1013          return "Hello " + name;
1014        }
1015
1016        public void print(){
1017          this.printer.print(sayHello());
1018        }
1019      }
1020
1021    3)com.example > right-click > New > Interface
1022    4)interface name : Printer
1023
1024      package com.example;
1025
1026      public interface Printer{
1027        void print(String message);
1028      }
1029
1030    5)com.example > right-click > New > Class
1031    6)Class Name : StringPrinter
1032    7)Interfaces : com.example.Printer
1033
1034      package com.example;
1035
1036      public class StringPrinter implements Printer{
1037        private StringBuffer buffer = new StringBuffer();
1038
1039        @Override
```

```
        public void print(String message){
          this.buffer.append(message);
        }

        public String toString(){
          return this.buffer.toString();
        }
      }
```

8)com.example > right-click > New > Class
9)Class Name : ConsolePrinter
10)Intefaces : com.example.Printer

```
    package com.example;

    public class ConsolePrinter implements Printer{

        @Override
        public void print(String message){
          System.out.println(message);
        }
      }
```

4. Java Project를 Spring Project로 변환
1)DIDemo2 Project > right-click > Configure > Convert to Maven Project
-Project : /DIDemo2
-Group Id : DIDemo2
-Artifact Id : DIDemo2
-version : 0.0.1-SNAPSHOT
-Packaging : jar
-Finish

2)DIDemo2 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>5.2.5.RELEASE</version>
      </dependency>
    </dependencies>
```

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

5. DIDemo2/resources folder 생성
1)DIDemo2 project > right-click > Build Path > Configure Build Path
2)Source Tab > Add Folder
3)DIDemo2 선택확인
4)Create New Folder > Folder name : resources > Finish > OK
5)DIDemo2/resources(new) 확인
6)Apply and Close

6. Bean Configuration XML 작성
1)DIDemo2/resources > right-click > New > Other > Spring > Spring Bean Configuration File
-File name : beans.xml > Finish

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans.xsd">

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

</beans>
```

7. Test class 작성
1)/src > right-click > New > Package
2)Package Name : com.example.test
3)/src/com.example/test/HelloBeanTest.java

```
package com.example.test;

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

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

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

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

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

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

    System.out.println(hello == hello2);  //Singleton Pattern
  }
}
```

8. Test

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

9. /src/com.example.Hello 생성자 추가

```
public Hello(String name, Printer printer) {
  this.name = name;
  this.printer = printer;
}
```

10. /resources/beans.xml에 아래 Code 추가

```
<bean id="hello2" class="com.example.Hello">
  <constructor-arg index="0" value="Spring" />
  <constructor-arg index="1" ref="printer" />
</bean>
```

11. /src/com.example.test/HelloBeanTest.java 수정

```
...
//2. Hello Beans 가져오기
Hello hello = (Hello)context.getBean("hello2");
...
Hello hello2 = context.getBean("hello2", Hello.class);
...
```

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-13.0.2' and workspace compiler preferences]
3)Next
4)Uncheck [Create module-info.java file]
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;
```

```
1212
1213      public class Student {
1214        private String name;
1215        private int age;
1216        private int grade;
1217        private int classNum;
1218      }
1219
1220    2)com.example.StudentInfo.java
1221      package com.example;
1222
1223      public class StudentInfo {
1224        private Student student;
1225      }
1226
1227
1228  4. Java Project를 Spring Project로 변환
1229    1)SpringDemo1 Project > right-click > Configure > Convert to Maven Project
1230      -Project : /SpringDemo1
1231      -Group Id : SpringDemo1
1232      -Artifact Id : SpringDemo1
1233      -version : 0.0.1-SNAPSHOT
1234      -Packaging : jar
1235      -Finish
1236
1237    2)SpringDemo1 Project > right-click > Spring > Add Spring Project Nature
1238
1239    3)pom.xml file에 Spring Context Dependency 추가하기
1240      <version>0.0.1-SNAPSHOT</version>
1241      <dependencies>
1242        <dependency>
1243          <groupId>org.springframework</groupId>
1244          <artifactId>spring-context</artifactId>
1245          <version>5.2.5.RELEASE</version>
1246        </dependency>
1247      </dependencies>
1248
1249    4)pom.xml > right-click > Run As > Maven install
1250      [INFO] BUILD SUCCESS 확인
1251
1252
1253  5. Lombok library 추가
1254    1)https://mvnrepository.com/에서 'lombok'으로 검색
1255    2)'Project Lombok' click
1256    3)1.18.12 click
1257    4)depency copy해서 pom.xml에 붙여넣기
1258
1259      <dependencies>
1260        <dependency>
1261          <groupId>org.springframework</groupId>
1262          <artifactId>spring-context</artifactId>
1263          <version>5.2.5.RELEASE</version>
1264        </dependency>
1265        <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
1266        <dependency>
1267          <groupId>org.projectlombok</groupId>
1268          <artifactId>lombok</artifactId>
1269          <version>1.18.12</version>
```

```
        <scope>provided</scope>
      </dependency>
    </dependencies>
```

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

6. Student.java와 StudentInfo.java 수정
1)Student.java

```
package com.example;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import lombok.AllArgsConstructor;

@Getter
@Setter
@ToString
@AllArgsConstructor
public class Student {
  private String name;
  private int age;
  private int grade;
  private int classNum;
}
```

2)StudentInfo.java

```
package com.example;

import lombok.Setter;
import lombok.AllArgsConstructor;

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

  public void printInfo(){
    if(this.student != null){
      System.out.println("Name : " + this.student.getName());
      System.out.println("Age : " + this.student.getAge());
      System.out.println("Grade : " + this.student.getGrade());
      System.out.println("Class : " + this.student.getClassNum());
      System.out.println("------------------------");
    }
  }
}
```

7. 환경설정을 위해 config package 생성
1)com.example package > right-click > New > Package
2)Name : com.example.config
3)Finish

8. ApplicationContext.java 생성
1)com.example.config > right-click > New > Class
2)Name : ApplicationCtx
3)Finish

```
package com.example.config;

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

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

@Configuration
public class ApplicationCtx {
  @Bean
  public Student student1() {
    return new Student("한지민", 15, 2, 5);
  }

  @Bean
  public Student student2() {
    return new Student("김지민", 16, 3,7);
  }

  @Bean
  public StudentInfo studentInfo() {
    return new StudentInfo(this.student1());
  }
}
```

9. com.example.MainClass.java

```
package com.example;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.example.config.ApplicationCtx;

public class MainClass {
  public static void main(String[] args) {
    ApplicationContext ctx = new
    AnnotationConfigApplicationContext(ApplicationCtx.class);

    StudentInfo studentInfo = ctx.getBean("studentInfo", StudentInfo.class);
    studentInfo.printInfo();

    Student student2 = ctx.getBean("student2", Student.class);
    studentInfo.setStudent(student2);
    studentInfo.printInfo();
  }
}
```

10. Java Application 실행

```
   Name : 한지민
   Age : 15
   Grade : 2
   Class : 5
   ------------------------
   Name : 김지민
   Age : 16
   Grade : 3
   Class : 7
   ------------------------



---------------------------------------
Task 10. Context file 여러개 사용하기
1. In Package Explorer > right-click > New > Java Project
   1)Project Name : SpringDemo2
   2)JRE
      -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
   3)Next
   4)Uncheck [Create module-info.java file]
   5)Finish


2)src > right-click > New > Package
   2)Package name : com.example


3. POJO Class 생성
   1)com.example.Student.java

      package com.example;

      import java.util.ArrayList;

      public class Student {
         private String name;
         private int age;
         private ArrayList<String> hobbys;
         private double height;
         private double weight;
      }

   2)com.example.StudentInfo.java

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

   3)com.example.Product.java

      package com.example;
      public class Product {
         private String pName;
         private int pPrice;
         private String maker;
         private String color;
```

```
    }


4. Java Project를 Spring Project로 변환
  1)SpringDemo2 Project > right-click > Configure > Convert to Maven Project
    -Project : /SpringDemo2
    -Group Id : SpringDemo2
    -Artifact Id : SpringDemo2
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish

  2)SpringDemo2 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>5.2.5.RELEASE</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)1.18.12 click
  4)depency copy해서 pom.xml에 붙여넣기

    <dependencies>
      <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.2.5.RELEASE</version>
      </dependency>
      <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
      <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.12</version>
        <scope>provided</scope>
      </dependency>
    </dependencies>

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


6. SpringDemo2/resources folder 생성
  1)SpringDemo2 project > right-click > new > Source Folder
  2)Folder Name : resources
  3)Finish

```

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

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

```
package com.example;

import java.util.ArrayList;

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

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

  2)StudentInfo.java

```
package com.example;

import lombok.Setter;
import lombok.Getter;

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

  3)Product.java

```
package com.example;

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

@NoArgsConstructor
@AllArgsConstructor
@RequiredArgsConstructor
```

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


9. applicationContext.xml
  <?xml version="1.0" encoding="UTF-8"?>
  <beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

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

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


10. applicationContext2.xml
  1)Namespace tab을 선택하여 c, p를 선택한다.
    <?xml version="1.0" encoding="UTF-8"?>
    <beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:c="http://www.springframework.org/schema/c"
      xmlns:p="http://www.springframework.org/schema/p"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
      http://www.springframework.org/schema/beans/spring-beans.xsd">

      <bean id="student3" class="com.example.Student">
        <constructor-arg value="김지민" />
        <constructor-arg value="50" />
        <constructor-arg>
          <list>
            <value>노래부르기</value>
            <value>게임</value>
```

```
          </list>
        </constructor-arg>
        <property name="height" value="175" />
        <property name="weight">
          <value>75</value>
        </property>
      </bean>

      <bean id="product" class="com.example.Product" c:pName="Computer"
      c:pPrice="2000000" p:maker="Samsung">
        <property name="color" value="Yellow" />
      </bean>
    </beans>


11. com.example.MainClass
  package com.example;

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

  public class MainClass {
    public static void main(String[] args) {
      String configFile = "classpath:applicationContext.xml";
      String configFile1 = "classpath:applicationContext2.xml";
      AbstractApplicationContext context = new GenericXmlApplicationContext(configFile,
      configFile1);
      Student student1 = context.getBean("student1", Student.class);
      System.out.println(student1);

      StudentInfo studentInfo = context.getBean("studentInfo1", StudentInfo.class);
      Student student2 = studentInfo.getStudent();
      System.out.println(student2);
      if(student1.equals(student2)) System.out.println("Equals");
      else System.out.println("Different");

      Student student3 = context.getBean("student3", Student.class);
      System.out.println(student3);

      if(student1.equals(student3)) System.out.println("Equals");
      else System.out.println("Different");

      Product product = context.getBean("product", Product.class);
      System.out.println(product);
      context.close();
    }
  }

12. Java Application 실행
  Student [name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,weight=45.0]
  Student [name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,weight=45.0]
  Equals
  Student [name=김지민, age=50, hobbys=[노래부르기, 게임], height=175.0,weight=75.0]
  Different
  Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]
```

Task 11. Java Annotation을 이용하여 두 개 이상의 설정 파일로 DI 설정하기
1. In Package Explorer > right-click > New > Java Project
  1)Project Name : SpringDemo3
  2)JRE
    -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
  3)Next
  4)Uncheck [Create module-info.java file]
  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;

import java.util.ArrayList;

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

  2)com.example.StudentInfo.java

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

  3)com.example.Product.java

```
package com.example;
public class Product {
   private String pName;
   private int pPrice;
   private String maker;
   private String color;
}
```

4. Java Project를 Spring Project로 변환
  1)SpringDemo3 Project > right-click > Configure > Convert to Maven Project
    -Project : /SpringDemo3
    -Group Id : SpringDemo3
    -Artifact Id : SpringDemo3
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish

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

```
1729
1730    3)pom.xml file에 Spring Context Dependency 추가하기
1731      <version>0.0.1-SNAPSHOT</version>
1732      <dependencies>
1733        <dependency>
1734          <groupId>org.springframework</groupId>
1735          <artifactId>spring-context</artifactId>
1736          <version>5.2.5.RELEASE</version>
1737        </dependency>
1738      </dependencies>
1739
1740    4)pom.xml > right-click > Run As > Maven install
1741      [INFO] BUILD SUCCESS 확인
1742
1743
1744  5. Lombok library 추가
1745    1)https://mvnrepository.com/에서 'lombok'으로 검색
1746    2)'Project Lombok' click
1747    3)1.18.12 click
1748    4)depency copy해서 pom.xml에 붙여넣기
1749
1750      <dependencies>
1751        <dependency>
1752          <groupId>org.springframework</groupId>
1753          <artifactId>spring-context</artifactId>
1754          <version>5.2.5.RELEASE</version>
1755        </dependency>
1756        <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
1757        <dependency>
1758          <groupId>org.projectlombok</groupId>
1759          <artifactId>lombok</artifactId>
1760          <version>1.18.12</version>
1761          <scope>provided</scope>
1762        </dependency>
1763      </dependencies>
1764
1765    5)pom.xml > right-click > Run As > Maven install
1766      [INFO] BUILD SUCCESS 확인
1767
1768
1769  6. com.example.config package 생성
1770    1)com.example > right-click > new > Package
1771    2)Name : com.example.config
1772    3)Finish
1773
1774
1775  7. 2개의 Config Class 작성
1776    1)com.example.config > right-click > New > Class
1777    2)Name : AppConfig1
1778    3)Finish
1779    4)com.example.config > right-click > New > Class
1780    5)Name : AppConfig2
1781    6)Finish
1782
1783
1784  8. Student.java, StudentInfo.java 그리고 Product.java에 lombok Annotation 붙이기
1785    1)Student.java
1786
```

```
1787      package com.example;
1788
1789      import java.util.List;
1790
1791      import lombok.AllArgsConstructor;
1792      import lombok.Data;
1793      import lombok.NonNull;
1794      import lombok.RequiredArgsConstructor;
1795
1796      @Data
1797      @RequiredArgsConstructor
1798      @AllArgsConstructor
1799      public class Student {
1800        private @NonNull String name;
1801        private @NonNull int age;
1802        private @NonNull List<String> hobbys;
1803        private double height;
1804        private double weight;
1805      }
1806
1807    2)StudentInfo.java
1808
1809      package com.example;
1810
1811      import lombok.Setter;
1812      import lombok.Getter;
1813
1814      @Setter
1815      @Getter
1816      public class StudentInfo {
1817        private Student student;
1818      }
1819
1820    3)Product.java
1821
1822      package com.example;
1823
1824      import lombok.AllArgsConstructor;
1825      import lombok.NoArgsConstructor;
1826      import lombok.NonNull;
1827      import lombok.RequiredArgsConstructor;
1828      import lombok.Setter;
1829      import lombok.ToString;
1830
1831      @NoArgsConstructor
1832      @AllArgsConstructor
1833      @RequiredArgsConstructor
1834      @Setter
1835      @ToString
1836      public class Product {
1837        private @NonNull String pName;
1838        private @NonNull int pPrice;
1839        private @NonNull String maker;
1840        private String color;
1841      }
1842
1843
1844  9. AppConfig1.java
```

```
1845  package com.example.config;
1846
1847  import java.util.Arrays;
1848  import java.util.List;
1849
1850  import org.springframework.context.annotation.Bean;
1851  import org.springframework.context.annotation.Configuration;
1852
1853  import com.example.Student;
1854  import com.example.StudentInfo;
1855
1856  @Configuration
1857  public class AppConfig1 {
1858    @Bean
1859    public Student student1() {
1860      List<String> list = Arrays.asList("독서", "영화감상", "요리");
1861      Student student1 = new Student("한지민", 25, list);
1862      student1.setHeight(165);
1863      student1.setWeight(45);
1864      return student1;
1865    }
1866
1867    @Bean
1868    public StudentInfo studentInfo1() {
1869      StudentInfo studentInfo1 = new StudentInfo();
1870      studentInfo1.setStudent(this.student1());
1871      return studentInfo1;
1872    }
1873  }
1874
1875
1876 10. AppConfig2.java
1877  package com.example.config;
1878
1879  import java.util.Arrays;
1880  import java.util.List;
1881
1882  import org.springframework.context.annotation.Bean;
1883  import org.springframework.context.annotation.Configuration;
1884
1885  import com.example.Product;
1886  import com.example.Student;
1887
1888  @Configuration
1889  public class AppConfig2 {
1890    @Bean
1891    public Student student3() {
1892      List<String> list = Arrays.asList("노래부르기", "게임");
1893      Student student3 = new Student("김지민", 50, list);
1894      student3.setHeight(175);
1895      student3.setWeight(75);
1896      return student3;
1897    }
1898
1899    @Bean
1900    public Product product() {
1901      Product product = new Product("Computer", 2000000, "Samsung");
1902      product.setColor("Yellow");
```

```
        return product;
      }
    }


11. com.example.MainClass
    package com.example;

    import org.springframework.context.ApplicationContext;
    import org.springframework.context.annotation.AnnotationConfigApplicationContext;

    import com.example.config.AppConfig1;
    import com.example.config.AppConfig2;

    public class MainClass {
      public static void main(String[] args) {
        ApplicationContext context = new
        AnnotationConfigApplicationContext(AppConfig1.class, AppConfig2.class);
        Student student1 = context.getBean("student1", Student.class);
        System.out.println(student1);

        StudentInfo studentInfo = context.getBean("studentInfo1", StudentInfo.class);
        Student student2 = studentInfo.getStudent();
        System.out.println(student2);
        if(student1.equals(student2)) System.out.println("Equals");
        else System.out.println("Different");

        Student student3 = context.getBean("student3", Student.class);
        System.out.println(student3);

        if(student1.equals(student3)) System.out.println("Equals");
        else System.out.println("Different");

        Product product = context.getBean("product", Product.class);
        System.out.println(product);
      }
    }


12. Java Application 실행
    Student [name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,weight=45.0]
    Student [name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,weight=45.0]
    Equals
    Student [name=김지민, age=50, hobbys=[노래부르기, 게임], height=175.0,weight=75.0]
    Different
    Product [pName=Computer, pPrice=2000000, maker=Samsung, color=Yellow]



------------------------------------------------------------
Task 12. Java Annotation과 XML 을 이용한 DI 설정 방법 : XML file에 Java file을 포함시켜 사용하는 방법
1. In Package Explorer > right-click > New > Java Project
    1)Project Name : SpringDemo4
    2)JRE
      -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
    3)Next
    4)Uncheck [Create module-info.java file]
    5)Finish
```

2. src > right-click > New > Package
  1)Package name : com.example

3. POJO 생성
  1)com.example.Student.java
```
package com.example;

import java.util.ArrayList;

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

4. Java Project를 Spring Project로 변환
  1)SpringDemo4 Project > right-click > Configure > Convert to Maven Project
    -Project : /SpringDemo4
    -Group Id : SpringDemo4
    -Artifact Id : SpringDemo4
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish

  2)SpringDemo4 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>5.2.5.RELEASE</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)1.18.12 click
  4)depency copy해서 pom.xml에 붙여넣기

```
  <dependencies>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context</artifactId>
      <version>5.2.5.RELEASE</version>
    </dependency>
```

```
        <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
        <dependency>
          <groupId>org.projectlombok</groupId>
          <artifactId>lombok</artifactId>
          <version>1.18.12</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> hobbys;
      private double height;
      private double weight;
    }


7. com.example.ApplicationConfig.java
  package com.example;

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

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

  @Configuration
  public class ApplicationConfig {
    @Bean
    public Student student1(){
      List<String> list = Arrays.asList("독서", "영화감상", "요리");

      Student1 student1 = new Student("한지민", 25, list);
      student1.setHeight(165);
      student1.setWeight(45);

      return student1;
    }
```

```
2076    }
2077
2078
2079  8. SpringDemo4/resources folder 생성
2080    1)SpringDemo4 project > right-click > Build Path > Configure Build Path
2081    2)Source Tab > Add Folder
2082    3)SpringDemo4 선택 확인
2083    4)Create New Folder > Folder name : resources > Finish > OK
2084    5)SpringDemo4/resources(new) 확인
2085    6)Apply and Close
2086
2087
2088  9. Bean Configuration XML 작성
2089    1)SpringDemo4/resources > right-click > New > Spring Bean Configuration File
2090    2)File name : applicationContext.xml > Finish
2091
2092
2093  10. /resources/applicationContext.xml
2094    <?xml version="1.0" encoding="UTF-8"?>
2095    <beans xmlns="http://www.springframework.org/schema/beans"
2096      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2097      xmlns:context="http://www.springframework.org/schema/context"
2098      xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd">
2099
2100      <bean class="org.springframework.context.annotation.ConfigurationClassPostProcessor"
        />
2101      <bean class="com.example.ApplicationConfig" />
2102      <bean id="student3" class="com.example.Student">
2103        <constructor-arg value="박지민" />
2104        <constructor-arg value="50" />
2105        <constructor-arg>
2106          <list>
2107            <value>노래부르기</value>
2108            <value>게임</value>
2109          </list>
2110        </constructor-arg>
2111        <property name="height" value="175" />
2112        <property name="weight">
2113          <value>75</value>
2114        </property>
2115      </bean>
2116    </beans>
2117
2118
2119  11. com.example.MainClass.java
2120    package com.example;
2121
2122    import org.springframework.context.support.AbstractApplicationContext;
2123    import org.springframework.context.support.GenericXmlApplicationContext;
2124
2125    public class MainClass {
2126      public static void main(String[] args) {
2127        String configFile = "classpath:applicationContext.xml";
2128        AbstractApplicationContext context = new GenericXmlApplicationContext(configFile);
2129        Student student1 = context.getBean("student1", Student.class);
2130        System.out.println(student1);
2131
```

```
2132        Student student3 = context.getBean("student3", Student.class);
2133        System.out.println(student3);
2134      }
2135    }
2136
2137
2138  12. Java Application 실행
2139    Student [name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,weight=45.0]
2140    Student [name=박지민, age=50, hobbys=[노래부르기, 게임], height=175.0,weight=75.0]
2141
2142
2143  13. JUnit을 사용한 DI test class 작성하기
2144    1)com.example > right-click > New > JUnit Test Case
2145    2)Select [New JUnit 4 test]
2146    3)Name : HelloBeanJUnitTest
2147    4)Finish
2148    5)[New JUnit Test Case] 창에서 Select [Perform the follwing action:] > Add JUnit 4 library to
        the build path
2149    6)OK
2150
2151
2152  14. JUnit을 사용한 Test
2153    1)src/com.example > New > Class
2154      -Name : HelloBeanJUnitTest.java
2155
2156      package com.example;
2157
2158      import static org.junit.Assert.assertEquals;
2159      import static org.junit.Assert.assertSame;
2160
2161      import org.junit.Before;
2162      import org.junit.Test;
2163      import org.springframework.context.ApplicationContext;
2164      import org.springframework.context.support.GenericXmlApplicationContext;
2165
2166      public class HelloBeanJUnitTest {
2167        ApplicationContext context;
2168
2169        @Before
2170        public void init(){
2171          context = new GenericXmlApplicationContext("classpath:applicationContext.xml");
2172        }
2173
2174        @Test
2175        public void test1(){
2176          Student student1 = (Student)context.getBean("student1");
2177          assertEquals("한지민", student1.getName());
2178          System.out.println(student1);
2179        }
2180
2181        @Test
2182        public void test2(){
2183          Student student3 = context.getBean("student3", Student.class);
2184          System.out.println(student3);
2185
2186          Student student4 = (Student)context.getBean("student3");
2187          assertSame(student3, student4);
2188        }
```

```
2189      }
2190
2191    2)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test
2192      -JUnit 창에 Green Bar
2193        Student(name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,
          weight=45.0)
2194        Student(name=박지민, age=50, hobbys=[노래부르기, 게임], height=175.0, weight=75.0)
2195
2196
2197  15. Spring TestContext Framework을 이용한 Test
2198    1)Spring-Test library 설치
2199      -http://mvnrepository.com에서 'spring test'로 검색
2200      -검색 결과 목록에서 'Spring TestContext Framework' Click
2201      -version 목록에서 5.2.5.RELEASE Click
2202
2203    2)dependency 복사해서 pom.xml에 붙여넣기
2204      <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
2205      <dependency>
2206        <groupId>org.springframework</groupId>
2207        <artifactId>spring-test</artifactId>
2208        <version>5.2.5.RELEASE</version>
2209        <scope>test</scope>
2210      </dependency>
2211
2212    3)pom.xml > right-click > Maven Install
2213      [INFO] BUILD SUCCESS
2214
2215    4)Spring-Test를 사용할 HelloBeanJunitSpringTest.java 작성
2216      -src/com.example > New > Class
2217      -Name : HelloBeanJunitSpringTest
2218      -Finish
2219
2220        package com.example;
2221
2222        import static org.junit.Assert.assertEquals;
2223        import static org.junit.Assert.assertSame;
2224
2225        import org.junit.Test;
2226        import org.junit.runner.RunWith;
2227        import org.springframework.beans.factory.annotation.Autowired;
2228        import org.springframework.context.ApplicationContext;
2229        import org.springframework.test.context.ContextConfiguration;
2230        import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
2231
2232        @RunWith(SpringJUnit4ClassRunner.class)
2233        @ContextConfiguration(locations="classpath:applicationContext.xml")
2234        public class HelloBeanJunitSpringTest {
2235          @Autowired
2236          ApplicationContext context;
2237
2238          @Test
2239          public void test1() {
2240            Student student1 = this.context.getBean("student1", Student.class);
2241            assertEquals(25, student1.getAge());
2242            System.out.println(student1);
2243          }
2244
2245          @Test
```

```
2246          public void test2() {
2247            Student student3 = (Student)this.context.getBean("student3");
2248            Student student4 = this.context.getBean("student3", Student.class);
2249            assertSame(student3, student4);
2250            System.out.println(student4);
2251          }
2252        }
2253
2254      -right-click > Run As > JUnit Test
2255      -결과 -> JUnit View에 초록색 bar
2256
2257    5)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
2258      -해당 Project > right-click > Build Path > Libraries tab
2259      -spring-test-5.2.5.RELEASE.jar 선택 후 [Remove] 로 삭제
2260      -Classpath 선택
2261      -[Add External JARs...] Click
2262      -Local M2 Repository(e.g
          C:\Users\bluee\.m2\repository\org\springframework\spring-test\5.2.5.RELEASE)에서 직접
          jar(spring-test-5.2.5.RELEASE.jar)를 선택할 것
2263      -[Order and Export] tab에서 spring-test-5.2.5.RELEASE.jar 선택 후 [Up] button을 클릭
2264      -해당 DIDemo/src 바로 아래까지 올리고 [Apply and Close] Click
2265
2266
2267
2268 ----------------------------------------------------
2269 Task 13. Java Annotation과 XML 을 이용한 DI 설정 방법 : Java file에 XML file을 포함시켜 사용하는 방법
2270 1. In Package Explorer > right-click > New > Java Projectn
2271    1)Project Name : SpringDemo5
2272    2)JRE
2273      -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
2274    3)Next
2275    4)Uncheck [Create module-info.java file]
2276    5)Finish
2277
2278
2279 2. src > right-click > New > Package
2280    1)Package name : com.example
2281    2)Finish
2282
2283
2284 3. com.example.Student.java
2285    package com.example;
2286
2287    import java.util.List;
2288
2289    public class Student {
2290      private String name;
2291      private int age;
2292      private List<String> hobbys;
2293      private double height;
2294      private double weight;
2295    }
2296
2297
2298 4. Java Project를 Spring Project로 변환
2299    1)SpringDemo5 Project > right-click > Configure > Convert to Maven Project
2300      -Project : /SpringDemo5
2301      -Group Id : SpringDemo5
```

-Artifact Id : SpringDemo5
-version : 0.0.1-SNAPSHOT
-Packaging : jar
-Finish

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

3)pom.xml file에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.2.5.RELEASE</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)1.18.12 click
4)depency copy해서 pom.xml에 붙여넣기

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.2.5.RELEASE</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.12</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> hobbys;
      private double height;
      private double weight;
    }


7. SpringDemo5/resources folder 생성
  1)SpringDemo5 project > right-click > Build Path > Configure Build Path
  2)Source Tab > Add Folder
  3)SpringDemo5 선택 확인
  4)Create New Folder > Folder name : resources > Finish > OK
  5)SpringDemo5/resources(new) 확인
  6)Apply and Close


8. Bean Configuration XML 작성
  1)SpringDemo5/resources > right-click > New > Spring Bean Configuration File
  2)File name : applicationContext.xml > Finish


9. /resources/applicationContext.xml
  <?xml version="1.0" encoding="UTF-8"?>
  <beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="student3" class="com.example.Student">
      <constructor-arg value="홍지민" />
      <constructor-arg value="30" />
      <constructor-arg>
        <list>
          <value>등산</value>
          <value>게임</value>
          <value>독서</value>
        </list>
      </constructor-arg>
      <property name="height" value="165" />
      <property name="weight">
        <value>49</value>
      </property>
    </bean>
  </beans>


10. com.example.ApplicationConfig.java
  package com.example;

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

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

```
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.ImportResource;

@Configuration
@ImportResource("classpath:ApplicationContext.xml")
public class ApplicationConfig {
  @Bean
  public Student student1(){
    List<String> hobbys = Arrays.asList("독서", "영화감상", "요리");

    Student student = new Student("한지민", 25, hobbys);
    student.setHeight(165);
    student.setWeight(45);

    return student;
  }
}
```

11. com.example.MainClass.java

```
package com.example;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class MainClass {
  public static void main(String[] args) {
    AnnotationConfigApplicationContext context = new
    AnnotationConfigApplicationContext(ApplicationConfig.class);
    Student student1 = context.getBean("student1", Student.class);
    System.out.println(student1);

    Student student3 = context.getBean("student3", Student.class);
    System.out.println(student3);

    context.close();
  }
}
```

12. Java Application 실행

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

13. JUnit 5를 사용한 DI test class 작성하기

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

14. pom.xml에 dependency 추가

1)JUnit 5 설치
-http://mvnrepository.com에서 'junit'로 검색
-검색 결과 목록에서 'JUnit Jupiter API' Click

-version 목록에서 5.6.2 click

2)dependency 복사해서 pom.xml에 붙여넣기

```
<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.6.2</version>
  <scope>test</scope>
</dependency>
```

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

[INFO] BUILD SUCCESS

-만일 ERROR 발생하면 다음과 같이 조치한다.

-SpringDemo5 > right-click > Maven > Update Project

-SpringDemo5가 check되어 있음을 확인하고 OK

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

[INFO] BUILD SUCCESS

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 {
  ApplicationContext context;

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

  @Test
  public void test1(){
    Student student1 = (Student)context.getBean("student1");
    assertEquals("한지민", 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

```
2531      -JUnit 창에 Green Bar
2532        Student(name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0,
            weight=45.0)
2533        Student(name=홍지민, age=30, hobbys=[등산, 게임, 독서], height=165.0, weight=49.0)
2534
2535
2536
2537 ----------------------------------------
2538 Task 14. Lab
2539 1. In Package Explorer > right-click > New > Java Project
2540   1)Project name : DIDemo3
2541   2)JRE
2542     -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
2543   3)Next
2544   4)Uncheck [Create module-info.java file]
2545   5)Finish
2546
2547
2548 2. src > right-click > New > Package
2549   1)Package name : com.example
2550   2)Finish
2551
2552
2553 3. POJO class 작성
2554   1)com.example > right-click > New > Class
2555   2)Class Name : Hello
2556
2557     package com.example;
2558
2559     public class Hello{
2560       private String name;
2561       private Printer printer;
2562
2563       public String sayHello(){
2564         return "Hello " + name;
2565       }
2566
2567       public void print(){
2568         this.printer.print(sayHello());
2569       }
2570     }
2571
2572   3)com.example > right-click > New > Interface
2573   4)interface name : Printer
2574
2575     package com.example;
2576
2577     public interface Printer{
2578       void print(String message);
2579     }
2580
2581   5)com.example > right-click > New > Class
2582   6)Class Name : StringPrinter
2583   7)Interfaces : com.example.Printer
2584
2585     package com.example;
2586
2587     public class StringPrinter implements Printer{
```

```
2588        private StringBuffer buffer = new StringBuffer();
2589
2590        @Override
2591        public void print(String message){
2592          this.buffer.append(message);
2593        }
2594
2595        public String toString(){
2596          return this.buffer.toString();
2597        }
2598      }
2599
2600    8)com.example > right-click > New > Class
2601    9)Class Name : ConsolePrinter
2602    10)Interfaces : com.example.Printer
2603
2604      package com.example;
2605
2606      public class ConsolePrinter implements Printer{
2607
2608        @Override
2609        public void print(String message){
2610          System.out.println(message);
2611        }
2612      }
2613
2614
2615  4. Java Project를 Spring Project로 변환
2616    1)DIDemo3 Project > right-click > Configure > Convert to Maven Project
2617      -Project : /DIDemo3
2618      -Group Id : DIDemo3
2619      -Artifact Id : DIDemo3
2620      -version : 0.0.1-SNAPSHOT
2621      -Packaging : jar
2622      -Finish
2623
2624    2)DIDemo3 Project > right-click > Spring > Add Spring Project Nature
2625
2626    3)pom.xml file에 Spring Context Dependency 추가하기
2627      <version>0.0.1-SNAPSHOT</version>
2628      <dependencies>
2629        <dependency>
2630          <groupId>org.springframework</groupId>
2631          <artifactId>spring-context</artifactId>
2632          <version>5.2.5.RELEASE</version>
2633        </dependency>
2634      </dependencies>
2635
2636    4)pom.xml > right-click > Run As > Maven install
2637      [INFO] BUILD SUCCESS 확인
2638
2639
2640  5. Lombok library 추가
2641    1)https://mvnrepository.com/에서 'lombok'으로 검색
2642    2)'Project Lombok' click
2643    3)1.18.12 click
2644    4)depency copy해서 pom.xml에 붙여넣기
2645
```

```
    <dependencies>
      <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>5.2.5.RELEASE</version>
      </dependency>
      <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
      <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.12</version>
        <scope>provided</scope>
      </dependency>
    </dependencies>

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


6. Hello.java에 lombok Annotation으로 수정하기

  package com.example;

  import lombok.NoArgsConstructor;
  import lombok.Setter;

  @Setter
  @NoArgsConstructor
  public class Hello {
    private String name;
    private Printer printer;

    public String sayHello(){
      return "Hello " + name;
    }

    public void print(){
      this.printer.print(sayHello());
    }
  }


7. src/config folder 생성
  1)/src > right-click > New > Folder
  2)Folder name : config


8. Bean Configuration XML 작성
  1)/src/config > right-click > New > Spring Bean Configuration File
  2)File name : beans.xml > Finish

    <?xml version="1.0" encoding="UTF-8"?>
    <beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
      http://www.springframework.org/schema/beans/spring-beans.xsd">

      <bean id="hello" class="com.example.Hello">
```

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

9. DI Test class 작성
  1)/src > right-click > New > Package
  2)Name : com.example.test
  3)Finish
  4)/src/com.example.test > right-click > New > Class
  5)Name : HelloBeanTest

```
  package com.example.test;

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

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

  public class HelloBeanTest {
    public static void main(String [] args){
      ApplicationContext context = new
      GenericXmlApplicationContext("config/beans.xml");

      Hello hello = (Hello)context.getBean("hello");
      System.out.println(hello.sayHello());
      hello.print();

      Printer printer = (Printer)context.getBean("printer");
      System.out.println(printer.toString());

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

      System.out.println(hello == hello2);  //Singleton Pattern
    }
  }
```

  6)Java Application 실행

```
  Hello Spring
  Hello Spring
  true
```

10. JUnit 5 Library 설치
  1)JUnit 5 설치
    -http://mvnrepository.com에서 'junit'로 검색
    -검색 결과 목록에서 'JUnit Jupiter API' Click
    -version 목록에서 5.6.2 click

  2)dependency 복사해서 pom.xml에 붙여넣기

```
  <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
  <dependency>
    <groupId>org.junit.jupiter</groupId>
```

```
        <artifactId>junit-jupiter-api</artifactId>
        <version>5.6.2</version>
        <scope>test</scope>
      </dependency>

  3)pom.xml > right-click > Maven Install
    [INFO] BUILD SUCCESS
    -만일 ERROR 발생하면 다음과 같이 조치한다.
    -SpringDemo5 > right-click > Maven > Update Project
    -SpringDemo5가 check되어 있음을 확인하고 OK
    -다시 pom.xml > right-click > Maven Install
      [INFO] BUILD SUCCESS


11. JUnit 5를 사용한 Test
  1)com.example.test > right-click > New > Class
  2)Name : HelloBeanJUnitTest

    package com.example.test;

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

    import org.junit.jupiter.api.BeforeEach;
    import org.junit.jupiter.api.Test;
    import org.springframework.context.ApplicationContext;
    import org.springframework.context.support.GenericXmlApplicationContext;

    import com.example.Hello;

    public class HelloBeanJUnitTest {
      ApplicationContext context;

      @BeforeEach
      public void init() {
        this.context = new GenericXmlApplicationContext("config/beans.xml");
      }

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

      @Test
      public void test2(){
        Hello hello = (Hello)context.getBean("hello");
        Hello hello2 = context.getBean("hello", Hello.class);
        assertSame(hello, hello2);
      }
    }


  3)HelloBeanJUnitTest.java > right-click > Run As > JUnit Test
    -JUnit 창에 Green Bar
```

12. Spring TestContext Framework
1)Spring-Test library 설치
2)pom.xml 수정

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-test</artifactId>
  <version>5.2.5.RELEASE</version>
  <scope>test</scope>
</dependency>
```

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

4)Spring-Test를 사용할 DI test class-HelloBeanJUnitSpringTest.java 작성하기
-/src/com.example.test > New > Class
-Name : HelloBeanJUnitSpringTest
-Finish

```
package com.example.test;

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

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

import com.example.Hello;

@ExtendWith(SpringExtension.class)
//JUnit 5.x에서 사용
@ContextConfiguration(locations="classpath:config/beans.xml")
public class HelloBeanJUnitSpringTest {
  @Autowired
  ApplicationContext context;

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

  @Test
  public void test2(){
    Hello hello = (Hello)context.getBean("hello");
    Hello hello2 = context.getBean("hello", Hello.class);
    assertSame(hello, hello2);
  }
}
```

5)right-click > Run As > Junit Test
6)결과 -> Junit View에 초록색 bar

7)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
-해당 Project > right-click > Build Path > Configure Build Path > Libraries tab
-spring-test-5.2.5.RELEASE.jar 선택 후 [Remove] 로 삭제
-Classpath 선택
-[Add External JARs...] Click
-Local M2 Repository(e.g C:\Users\사용자아이디\.m2\repository\org\springframework\spring-test\5.2.5.RELEASE)에서 직접 jar(spring-test-5.2.5.RELEASE.jar)를 선택할 것
-[Order and Export] tab에서 spring-test-5.2.5.RELEASE.jar 선택 후 [Up] button을 클릭
-해당 Project/src 바로 아래까지 올리고 [Apply and Close] Click

13. src/com.example/StringPrinter.java 수정

```
package com.example;

import org.springframework.stereotype.Component;

@Component("stringPrinter")
public class StringPrinter implements Printer{
  private StringBuffer buffer = new StringBuffer();
...
```

14. src/com.example/ConsolePrinter.java 수정

```
package com.example;

import org.springframework.stereotype.Component;

@Component("consolePrinter")
public class ConsolePrinter implements Printer{
...
```

15. /src/com.example/Hello.java 수정

```
package com.example;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

import lombok.NoArgsConstructor;
import lombok.Setter;

@Setter
@NoArgsConstructor
@Component
public class Hello {
  @Value("Spring")
  private String name;

  @Autowired
  @Qualifier("stringPrinter")
  private Printer printer;

  public String sayHello(){
    return "Hello " + name;
```

```
    }

    public void print(){
      this.printer.print(sayHello());
    }
  }
```

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

17. 새로운 설정 file 생성
  1)src/config > right-click > New > Spring Bean Configuration File
  2)File name : annos.xml > Finish
  3)Namespace tab > context Check

```
  <?xml version="1.0" encoding="UTF-8"?>
  <beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
      http://www.springframework.org/schema/context
      http://www.springframework.org/schema/context/spring-context-4.3.xsd">

    <context:component-scan base-package="com.example" />
  </beans>
```

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

```
    package com.example.test;

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

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

    import com.example.Hello;

    @ExtendWith(SpringExtension.class)
    @ContextConfiguration(locations="classpath:config/annos.xml")
    public class HelloBeanJUnitSpringTest {
      @Autowired
      ApplicationContext context;

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

      @Test
```

```
2988        public void test2(){
2989          Hello hello = (Hello)context.getBean("hello");
2990          Hello hello2 = context.getBean("hello", Hello.class);
2991          assertSame(hello, hello2);
2992        }
2993      }
2994
2995    1)right-click > Run As > Junit Test
2996    2)결과 -> Junit View에 초록색 bar
2997
2998
2999
3000  ------------------------------
3001  Task 15. Lab with JUnit 5 Jupiter
3002  1. In Package Explorer > right-click > New > Java Project
3003    1)Project Name : DIDemo4
3004    2)JRE
3005      -Select [Use default JRE 'jdk-13.0.2' and workspace compiler preferences]
3006    3)Next
3007    4)Uncheck [Create module-info.java file]
3008    5)Finish
3009
3010
3011  2. src > right-click > New > Package
3012    1)Package name : com.example
3013    2)Finish
3014
3015
3016  3. com.example.Student.java, com.example.StudentInfo.java
3017    1)Student.java
3018      package com.example;
3019
3020      import java.util.List;
3021
3022      public class Student {
3023        private String name;
3024        private int age;
3025        private List<String> hobbys;
3026        private double height;
3027        private double weight;
3028      }
3029
3030    2)StudentInfo.java
3031      package com.example;
3032
3033      public class StudentInfo {
3034        private Student student;
3035      }
3036
3037
3038  4. Java Project를 Spring Project로 변환
3039    1)DIDemo4 Project > right-click > Configure > Convert to Maven Project
3040      -Project : /DIDemo4
3041      -Group Id : DIDemo4
3042      -Artifact Id : DIDemo4
3043      -version : 0.0.1-SNAPSHOT
3044      -Packaging : jar
3045      -Finish
```

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

3)pom.xml file에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.2.5.RELEASE</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)1.18.12 click
4)depency copy해서 pom.xml에 붙여넣기

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.2.5.RELEASE</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.12</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```

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

6. Student.java, StudentInfo.java lombok Annotation 붙이기
1)Student.java

```
package com.example;

import java.util.List;

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

@Data
@RequiredArgsConstructor
@AllArgsConstructor
public class Student {
```

```
3104        private @NonNull String name;
3105        private @NonNull int age;
3106        private @NonNull List<String> hobbys;
3107        private double height;
3108        private double weight;
3109      }
3110
3111    2)StudentInfo.java
3112
3113      package com.example;
3114
3115      import lombok.AllArgsConstructor;
3116      import lombok.Setter;
3117
3118      @Setter
3119      @AllArgsConstructor
3120      public class StudentInfo {
3121        private Student student;
3122
3123        public void printInfo(){
3124          if(this.student != null){
3125            System.out.println("Name : " + this.student.getName());
3126            System.out.println("Age : " + this.student.getAge());
3127            System.out.println("Hobbies");
3128            this.student.getHobbys().forEach(hobby -> System.out.println(hobby));
3129            System.out.println("Height : " + this.student.getHeight());
3130            System.out.println("Weight : " + this.student.getWeight());
3131          }
3132        }
3133      }
3134
3135
3136  7. com.example.config package 생성
3137    1)com.example > right-click > New > Package
3138    2)Name : com.example.config
3139    3)Finish
3140
3141
3142  8. com.example.config.ApplicationConfig.java 생성
3143    1)com.example.config > right-click > New > Click
3144    2)Name : ApplicationConfig
3145    3)Finish
3146
3147      package com.example.config;
3148
3149      import java.util.Arrays;
3150      import java.util.List;
3151
3152      import org.springframework.context.annotation.Bean;
3153      import org.springframework.context.annotation.Configuration;
3154
3155      import com.example.Student;
3156      import com.example.StudentInfo;
3157
3158      @Configuration
3159      public class ApplicationConfig {
3160        @Bean
3161        public Student student1() {
```

```
3162          List<String> list = Arrays.asList("독서", "영화감상", "요리");
3163          Student student1 = new Student("한지민", 25, list);
3164          student1.setHeight(165);
3165          student1.setWeight(45);
3166          return student1;
3167        }
3168
3169        @Bean
3170        public StudentInfo studentInfo() {
3171          return new StudentInfo(this.student1());
3172        }
3173      }
3174
3175
3176  9. com.example.MainClass.java
3177
3178    package com.example;
3179
3180    import org.springframework.context.annotation.AnnotationConfigApplicationContext;
3181
3182    import com.example.config.ApplicationConfig;
3183
3184    public class MainClass {
3185      public static void main(String[] args) {
3186        AnnotationConfigApplicationContext context = new
          AnnotationConfigApplicationContext(ApplicationConfig.class);
3187        Student student1 = context.getBean("student1", Student.class);
3188        System.out.println(student1);
3189
3190        StudentInfo studentInfo = context.getBean("studentInfo", StudentInfo.class);
3191        studentInfo.setStudent(student1);
3192        studentInfo.printInfo();
3193
3194        context.close();
3195      }
3196    }
3197
3198
3199  10. Java Application 실행
3200    Student(name=한지민, age=25, hobbys=[독서, 영화감상, 요리], height=165.0, weight=45.0)
3201    Name : 한지민
3202    Age : 25
3203    Hobbies
3204    독서
3205    영화감상
3206    요리
3207    Height : 165.0
3208    Weight : 45.0
3209
3210
3211  11. Student.java 수정
3212
3213    package com.example;
3214
3215    import java.util.List;
3216
3217    import org.springframework.beans.factory.annotation.Value;
3218    import org.springframework.stereotype.Component;
```

```
3219
3220    import lombok.Getter;
3221    import lombok.Setter;
3222
3223    @Component
3224    @Setter
3225    @Getter
3226    public class Student {
3227      @Value("한지민")
3228      private String name;
3229      @Value("25")
3230      private int age;
3231      @Value("등산, 게임, 독서")
3232      private List<String> hobbys;
3233      @Value("162.5")
3234      private double height;
3235      @Value("49.2")
3236      private double weight;
3237    }
3238
3239
3240  12. StudentInfo.java 수정
3241
3242    package com.example;
3243
3244    import org.springframework.beans.factory.annotation.Autowired;
3245    import org.springframework.stereotype.Component;
3246
3247    import lombok.NoArgsConstructor;
3248    import lombok.Setter;
3249
3250    @NoArgsConstructor
3251    @Component
3252    public class StudentInfo {
3253      @Setter(onMethod_ = @Autowired)
3254      private Student student;
3255
3256      public void printInfo(){
3257        if(this.student != null){
3258          System.out.println("Name : " + this.student.getName());
3259          System.out.println("Age : " + this.student.getAge());
3260          System.out.println("Hobbies");
3261          this.student.getHobbys().forEach(hobby -> System.out.println(hobby));
3262          System.out.println("Height : " + this.student.getHeight());
3263          System.out.println("Weight : " + this.student.getWeight());
3264        }else {
3265          System.out.println("Null");
3266        }
3267      }
3268    }
3269
3270
3271  13. ApplicationConfig.java 수정
3272
3273    package com.example.config;
3274
3275    import org.springframework.context.annotation.Bean;
3276    import org.springframework.context.annotation.ComponentScan;
```

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

import com.example.StudentInfo;

@Configuration
@ComponentScan(basePackages = {"com.example"})
public class ApplicationConfig {
  @Bean
  public StudentInfo studentInfo() {
    return new StudentInfo();
  }
}
```

14. MainClass.java 수정

```
package com.example;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import com.example.config.ApplicationConfig;

public class MainClass {
  public static void main(String[] args) {
    AnnotationConfigApplicationContext context = new
    AnnotationConfigApplicationContext(ApplicationConfig.class);
    StudentInfo info = context.getBean("studentInfo", StudentInfo.class);
    info.printInfo();
    context.close();
  }
}
```

15. MainClass 실행

```
Name : 한지민
Age : 25
Hobbies
등산, 게임, 독서
Height : 162.5
Weight : 49.2
```