```
HOL : 외부 파일을 이용한 설정
----------------------------------------------
Task1. Lab
1. In Package Explorer > right-click > New > Java Project
  1)Project Name : EnvironmentDemo
  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)Name : com.example
  2)Finish


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

  2)EnvironmentDemo 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


4. 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

5. EnvironmentDemo/resources folder 생성

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

  2)Folder name : resources

  3)Finish

6. resources > right-click > New > File

  1)File name : admin.properties

  2)Finish

```
admin.id=javaexpert
admin.pwd=12345678
```

7. com.example.AdminConnection 생성

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

  2)Name : AdminConnection

  3)Finish

```
package com.example;

import org.springframework.beans.factory.DisposableBean;
import org.springframework.beans.factory.InitializingBean;
import org.springframework.context.EnvironmentAware;
import org.springframework.core.env.Environment;

import lombok.Getter;
import lombok.Setter;

public class AdminConnection implements EnvironmentAware, InitializingBean, DisposableBean {
   @Setter
   private Environment env;
   @Getter
   @Setter
   private String adminId;
   @Getter
   @Setter
   private String adminPwd;

   @Override
   public void destroy() throws Exception {
      System.out.println("destroy()");
   }

   @Override
   public void afterPropertiesSet() throws Exception {
      System.out.println("afterPropertiesSet()");
      setAdminId(env.getProperty("admin.id"));
```

```
            setAdminPwd(env.getProperty("admin.pwd"));
        }

        // bean이 생성되기 전에 callback 으로 호출됨. 가장 먼저 호출됨.
        // MainClass에서 사용하는 env 정보가 넘어옴.
        @Override
        public void setEnvironment(Environment env) {
            System.out.println("setEnvironment()");
            setEnv(env);
        }
    }
```

8. resources > right-click > New > Spring Bean Configuration File
  1)File name : beans.xml
  2)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="adminConnection" class="com.example.AdminConnection" />

  </beans>
```

9. com.example.MainClass Class 생성
  1)com.example > right-click > New > Class
  2)Name : MainClass
  3)Finish

```
    package com.example;

    import java.io.IOException;

    import org.springframework.context.ConfigurableApplicationContext;
    import org.springframework.context.support.GenericXmlApplicationContext;
    import org.springframework.core.env.ConfigurableEnvironment;
    import org.springframework.core.env.MutablePropertySources;
    import org.springframework.core.io.support.ResourcePropertySource;

    public class MainClass {
        public static void main(String [] args){
            ConfigurableApplicationContext ctx = new GenericXmlApplicationContext();
            ConfigurableEnvironment env = ctx.getEnvironment();

            MutablePropertySources propertySouces = env.getPropertySources();
            //내가 원하는 정보를 얻을 때까지 모든 propertySources를 앞에서 부터 차례로 모두 검색함.
            try{
                propertySouces.addLast(new ResourcePropertySource("classpath:admin.properties"));  //property 추가

                System.out.println(env.getProperty("admin.id"));    //property 추출
                System.out.println(env.getProperty("admin.pwd"));
            }catch(IOException ex){}

            GenericXmlApplicationContext gCtx = (GenericXmlApplicationContext)ctx;
            gCtx.load("classpath:beans.xml");
            gCtx.refresh();

            AdminConnection adminConnection = gCtx.getBean("adminConnection",
            AdminConnection.class);
            System.out.println("admin ID : " + adminConnection.getAdminId());
            System.out.println("admin PWD : " + adminConnection.getAdminPwd());

            gCtx.close();
            ctx.close();
         }
      }
```

10. 실행

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

```
javaexpert
12345678
setEnvironment()
afterPropertiesSet()
admin ID : javaexpert
admin PWD : 12345678
destroy()
```

Task2. Lab

1. In Package Explorer > right-click > New > Java Project
   1)Project Name : PropertyDemo
   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)Name : com.example
   2)Finish

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

```
package com.example;

import java.util.List;

public class Hello {
   private String name;
   private Printer printer;
   private List<String> names;

```

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

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

4)com.example > right-click > New > Interface

5)Name : Printer

6)Finish

```
  package com.example;

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

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

8)Name : StringPrinter

9)Interfaces : com.example.Printer

10)Finish

```
  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();
    }
  }
```

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

12)Name : ConsolePrinter

13)Interfaces : com.example.Printer

14)Finish

```
  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)PropertyDemo Project > right-click > Configure > Convert to Maven Project

-Project : /PropertyDemo

```
    -Group Id : PropertyDemo
    -Artifact Id : PropertyDemo
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish
```

2)PropertyDemo 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>
```

4pom.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. Hello.java에 lombok Annotation 붙이기

1)Hello.java

```
    package com.example;

    import java.util.List;

    import lombok.Getter;
    import lombok.NoArgsConstructor;
    import lombok.Setter;

```

```
@NoArgsConstructor
@Setter
public class Hello {
  private String name;
  private Printer printer;
  @Getter private List<String> names;

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

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

7. PropertyDemo/resources folder 생성
   1)PropertyDemo project > right-click > New > Source Folder
   2)Folder name : resources
   3)Finish

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

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

  <bean id="hello" class="com.example.Hello">
    <property name="name" value="Spring" />
    <property name="printer" ref="printer" />
    <property name="names">
      <list>
        <value>AOP</value>
        <value>Spring</value>
        <value>DI</value>
      </list>
    </property>
  </bean>

  <bean id="printer" class="com.example.StringPrinter" />
  <bean id="consolePrinter" class="com.example.ConsolePrinter" />

</beans>
```

9. com.example.MainClass Class 생성
   1)com.example > right-click > New > Class
   2)Name : MainClass
   3)Finish

```
package com.example;
```

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

public class MainClass {
  public static void main(String [] args){
    ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");

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

    Printer printer = ctx.getBean("printer", StringPrinter.class);
    System.out.println(printer.toString());

    hello.getNames().forEach(value -> System.out.println(value));
  }
}
```

10. 실행

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

```
Hello Spring
Hello Spring
AOP
Spring
DI
```

11. JUnit 5 & Spring TestContext Framework으로 Test

1)src > right-click > New > Package
2)Name : com.example.test
3)Finish
4)com.example.test > right-click > New > JUnit Test Case > Select [New JUnit Jupiter test]
5)Name : HelloJUnitTest
6)Finish
7)New JUnit Test Case창에서, [Perform the following action: Add JUnit 5 library to the build path]
8)OK
9)https://mvnrepository.com에서 'spring-test'로 검색
10)Spring TestContext Framework에서
11)5.2.5.RELEASE로 들어가서
12)복사해서 pom.xml로 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>
```

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

[INFO] BUILD SUCCESS 확인
-만일 Error 발생하면 PropertyDemo > right-click > Maven > Update Project > OK
-pom.xml > right-click > Run As > Maven install
-[INFO] BUILD SUCCESS 확인

12. com.example.test.HelloJUnitTest

```
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;
import com.example.Printer;

@ExtendWith(SpringExtension.class)
@ContextConfiguration(locations = "classpath:beans.xml")
class HelloJUnitTest {
   @Autowired
   ApplicationContext ctx;

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

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

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

      Hello hello2 = ctx.getBean("hello", Hello.class);
      assertSame(hello, hello2);

      assertEquals(3, hello2.getNames().size());
   }
}
```

1)만일 해당 객체를 찾을 수 없다는 오류가 계속 발생하면
   -해당 Project > right-click > Build Path > Libraries tab
   -spring-test-5.2.5.RELEASE.jar 선택 후 [Remove] 로 삭제
   -[Add External JARs...] Click
   -%M2_HOME%\repository\org\springframework\spring-test\5.2.5.RELEASE 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

2)right-click > Run As > Junit Test

3)결과 -> Junit View에 초록색 bar

13. resources/value.properties 생성

```
myname=Spring
myprinter=printer
value1=HTML5
value2=CSS3
value3=JavaScript
```

14. /resources/beans.xml 수정하기
  1)beans.xml에서 [Namespaces] tab
  2)목록에서 'context-http://www.springframework.org/schema/context' check
  3)<context:property-placeholder />를 사용하기 위해서 다음과 같이 수정

```
<context:property-placeholder
        location="classpath:value.properties" />

<bean id="hello" class="com.example.Hello">
  <property name="name" value="${myname}" />
  <property name="printer" ref="${myprinter}" />
  <property name="names">
    <list>
      <value>${value1}</value>
      <value>${value2}</value>
      <value>${value3}</value>
    </list>
  </property>
</bean>

<bean id="printer" class="com.example.StringPrinter" />
<bean id="consolePrinter" class="com.example.ConsolePrinter" />
```

15. Test
  1)com.example.MainClass.java
  2)right-click > Run As > Java Application

```
Hello Spring
Hello Spring
HTML5
CSS3
JavaScript
```

  3)/src/test/java/HelloJUnitTest.java
    -right-click > Run As > JUnit Test
    -Green Bar

16. resources/value.properties 수정

```
myname=Spring
myprinter=printer
value1=HTML5
value2=CSS3
value3=JavaScript
printer1=stringPrinter
printer2=consolePrinter
```

17. pom.xml에 javax.annotation API dependency 추가하기

1)https://mvnrepository.com에서 'javax annotation'로 검색

2)Javax Annotation API click

3)1.3.2. click

4)dependency 복사해서 pom.xml로 붙여넣기

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

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

```
[INFO] BUILD SUCCESS 확인
```

18. Hello.java code 수정

1)com.example/Hello.java

```
package com.example;

import java.util.List;

import javax.annotation.Resource;

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

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

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

    @Resource(name="${printer1}")
    private Printer printer;

    @Value("${value1},${value2},${value3}")
    @Getter private List<String> names;

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

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

19. StringPrinter.java 수정

```
package com.example;
```

```java
import org.springframework.stereotype.Component;

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

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

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

20. beans.xml 수정하기

```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"
  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-3.2.xsd">

  <context:property-placeholder location="classpath:value.properties "/>
  <context:component-scan base-package="com.example" />  <---추가하기
  <!-- 나머지 코드 삭제 -->
</beans>
```

21. MainClass.java 수정하기

```java
package com.example;

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

public class MainClass {
  public static void main(String[] args) {
    ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");

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

    Printer printer = ctx.getBean("stringPrinter", StringPrinter.class);
    System.out.println(printer.toString());

    hello.getNames().forEach(value -> System.out.println(value));
  }
}
```

```
22. 실행
  1)MainClass > right-click > Run As > Java Application

  2)결과
    Hello Spring
    Hello Spring
    HTML5,CSS3,JavaScript



--------------------------------------------
Task 3. Lab
1. In Package Explorer > right-click > New > Java Project
  1)Project Name : PropertyDemo1
  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. /src/com.example.AdminConnection.java 생성
  1)com.example > right-click > New > Class
  2)Name : AdminConnection
  3)Finish

    package com.example;

    public class AdminConnection {
      private String adminId;
      private String adminPwd;
      private String subAdminId;
      private String subAdminPwd;
    }


4. Java Project를 Spring Project로 변환
  1)PropertyDemo1 Project > right-click > Configure > Convert to Maven Project
    -Project : /PropertyDemo1
    -Group Id : PropertyDemo1
    -Artifact Id : PropertyDemo1
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar
    -Finish

  2)PropertyDemo1 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. AdminConnection.java 수정

```
    package com.example;

    import lombok.Getter;
    import lombok.Setter;

    @Setter
    @Getter
    public class AdminConnection {
      private String adminId;
      private String adminPwd;
      private String subAdminId;
      private String subAdminPwd;
    }
```

7. PropertyDemo/resources folder 생성
1)PropertyDemo1 project > right-click > New > Source Folder
2)Folder name : resources
3)Finish

8./resources 두 개의 properties file 생성

<admin.properties>

```
    admin.id=javaexpert
    admin.pwd=12345678

  <sub.admin.properties>
    sub.admin.id=javasoft
    sub.admin.pwd=987654321
```

9. Bean Configuration XML 작성

  1)PropertyDemo1/resources > right-click > New > Spring Bean Configuration File
  2)File name : beans.xml
  3)Finish
  4)Namespace tab에서 context - http://www.springframework.org/schema/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-3.2.xsd">

      <context:property-placeholder location="classpath:admin.properties, classpath:sub.admin.properties" />

      <bean id="adminConnection" class="com.example.AdminConnection">
        <property name="adminId">
          <value>${admin.id}</value>
        </property>
        <property name="adminPwd">
          <value>${admin.pwd}</value>
        </property>
        <property name="subAdminId">
          <value>${sub.admin.id}</value>
        </property>
        <property name="subAdminPwd">
          <value>${sub.admin.pwd}</value>
        </property>
      </bean>
    </beans>
```

10. /src/com.example.MainClass.java 생성

  1)com.example > right-click > New > Class
  2)Name : MainClass
  3)Finish

```
  package com.example;

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

  public class MainClass {
    public static void main(String[] args) {
      AbstractApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");
      AdminConnection connection = ctx.getBean("adminConnection",
      AdminConnection.class);
      System.out.println("admin ID : " + connection.getAdminId());
      System.out.println("admin PWD : " + connection.getAdminPwd());
      System.out.println("sub admin ID : " + connection.getSubAdminId());
      System.out.println("sub admin PWD : " + connection.getSubAdminPwd());

      ctx.close();
    }
  }
```

11. 결과

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

```
    admin ID : javaexpert
    admin PWD : 12345678
    sub admin ID : javasoft
    sub admin PWD : 987654321
```

Task 4. Lab

1. In Package Explorer > right-click > New > Java Project
  1)Project Name : PropertyDemo2
  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. /src/com.example.AdminConnection.java 생성
  1)com.example > right-click > New > Class
  2)Name : AdminConnection
  3)Finish

```
    package com.example;

    public class AdminConnection {
      private String adminId;
      private String adminPwd;
      private String subAdminId;
      private String subAdminPwd;
    }
```

4. Java Project를 Spring Project로 변환
  1)PropertyDemo1 Project > right-click > Configure > Convert to Maven Project
    -Project : /PropertyDemo2
    -Group Id : PropertyDemo2
    -Artifact Id : PropertyDemo2
    -version : 0.0.1-SNAPSHOT
    -Packaging : jar

-Finish

2)PropertyDemo2 Project > right-click > Spring > Add Spring Project Nature

3)pom.xml 파일에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.2.5.RELEASE</version>
  </dependency>
</dependencies>
```

4pom.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. AdminConnection.java 수정

```
package com.example;

import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
public class AdminConnection {
  private String adminId;
  private String adminPwd;
  private String subAdminId;
  private String subAdminPwd;
}
```

7. PropertyDemo/resources folder 생성
   1)PropertyDemo1 project > right-click > New > Source Folder
   2)Folder name : resources
   3)Finish

8. /resources 두 개의 properties file 생성

```
  <admin.properties>
    admin.id=javaexpert
    admin.pwd=12345678

  <sub.admin.properties>
    sub.admin.id=javasoft
    sub.admin.pwd=987654321
```

9. ApplicationConfig Class 생성
   1)com.example > right-click > New > Class
   2)Name : ApplicationConfig
   3)Finish

```java
    package com.example;

    import org.springframework.context.annotation.Bean;
    import org.springframework.context.annotation.ComponentScan;
    import org.springframework.context.annotation.Configuration;
    import org.springframework.context.support.PropertySourcesPlaceholderConfigurer;
    import org.springframework.core.io.ClassPathResource;

    @Configuration
    @ComponentScan(basePackages = "com.example")
    public class ApplicationConfig {
      @Bean
      public static PropertySourcesPlaceholderConfigurer properties() {
        PropertySourcesPlaceholderConfigurer configurer = new PropertySourcesPlaceholderConfigurer();
        configurer.setLocations(new ClassPathResource("admin.properties"), new ClassPathResource("sub.admin.properties"));
        return configurer;
      }
    }
```

10. com.example.AdminConnection.java 수정

```java
  package com.example;

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

  import lombok.Getter;

  @Getter
  @Component
  public class AdminConnection {
    @Value("${admin.id}")
    private String adminId;
    @Value("${admin.pwd}")
    private String adminPwd;
    @Value("${sub.admin.id}")
    private String subAdminId;
    @Value("${sub.admin.pwd}")
    private String subAdminPwd;
  }
```

11. /src/com.example.MainClass.java

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

  2)Name : MainClass1

  3)Finish

```
    package com.example;

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

    public class MainClass1 {
      public static void main(String[] args) {
        ApplicationContext context = new AnnotationConfigApplicationContext(ApplicationConfig.class);
        AdminConnection connection = (AdminConnection) context.getBean("adminConnection");
        System.out.println("admin ID : " + connection.getAdminId());
        System.out.println("admin PWD : " + connection.getAdminPwd());
        System.out.println("sub admin ID : " + connection.getSubAdminId());
        System.out.println("sub admin PWD : " + connection.getSubAdminPwd());
      }
    }
```

12. 결과

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

```
    admin ID : javaexpert
    admin PWD : 12345678
    sub admin ID : javasoft
    sub admin PWD : 987654321
```

Task 5. Lab

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

  1)Project Name : ProfileDemo

  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. Package 생성

  1)/src/ > right-click > New > Package

  2)Package name : com.example

3)Finish

3. Java Project를 Spring Project로 변환

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

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

2)ProfileDemo 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

4. 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

5. ProfileDemo/resources folder 생성

1)ProfileDemo project > New > Source Folder

2)Folder name : resources

3)Finish

6. ServerInfo.java 생성
  1)com.example > right-click > New >Class
  2)Name : ServerInfo
  3)Finish

```
package com.example;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class ServerInfo {
  private String ipNum;
  private String portNum;
}
```

7. XML 설정 file 2개 생성
  1)resource > right-click > New > Spring Bean Configuration File
  2)File name : run.xml
  3)Finish

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

  <bean id="serverInfo" class="com.example.ServerInfo">
    <property name="ipNum" value="192.168.56.5" />
    <property name="portNum" value="80" />
  </bean>
</beans>
```

  3)/resource > right-click > New > Spring Bean Configuration File
  4)File name : dev.xml
  5)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"
  profile="dev">    <---이것이 핵심

  <bean id="serverInfo" class="com.example.ServerInfo">
    <property name="ipNum" value="localhost" />
    <property name="portNum" value="8080" />
  </bean>
</beans>
```

8. MainClass 생성
  1)com.example > right-click > New > Class
  2)Name : MainClass.java

```
   3)Finish

      package com.example;

      import java.util.Scanner;

      import org.springframework.context.support.GenericXmlApplicationContext;

      public class MainClass {
         public static void main(String[] args) {
            Scanner scan = new Scanner(System.in);
            System.out.print("Select dev or run : ");
            String config = scan.next();  //"dev" or "run"

            GenericXmlApplicationContext ctx = new GenericXmlApplicationContext();
            ctx.getEnvironment().setActiveProfiles(config);
            ctx.load("dev.xml", "run.xml");
            ctx.refresh();

            ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
            System.out.println("IP : " + info.getIpNum());
            System.out.println("Port : " + info.getPortNum());
            ctx.close();
         }
      }


9. 결과
   1)MainClass.java > right-click > Run As > Java Application
      Select dev or run :
   2)입력시 dev를 넣으면 dev환경인 localhost/8080이 나오고, 만일 run이라고 넣으면 192.168.56.5/80이 나온다.



------------------------------------
Task 6. Lab
1. In Package Explorer > right-click > New > Java Project
   1)Project Name : ProfileDemo1
   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. Package 생성
   1)/src/ > right-click > New > Package
   2)Package name : com.example
   3)Finish


3. Java Project를 Spring Project로 변환
   1)ProfileDemo1 Project > right-click > Configure > Convert to Maven Project
      -Project : /ProfileDemo1
      -Group Id : ProfileDemo1
      -Artifact Id : ProfileDemo1
      -version : 0.0.1-SNAPSHOT
```

-Packaging : jar
-Finish

2)ProfileDemo1 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

4. 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

5. ServerInfo.java 생성
1)com.example > right-click > New >Class
2)Name : ServerInfo
3)Finish

```
package com.example;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class ServerInfo {
  private String ipNum;
```

```
   private String portNum;
}
```

6. Java 설정 file 2개 생성
  1)com.example > right-click > New > Class
  2)Name : ApplicationConfigDev
  3)Finish

```
package com.example;

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

@Configuration
@Profile("dev")
public class ApplicationConfigDev {

   @Bean
   public ServerInfo serverInfo(){
      ServerInfo info = new ServerInfo();
      info.setIpNum("localhost");
      info.setPortNum("8080");
      return info;
   }
}
```

  4)com.example > right-click > New > Class
  5)Name : ApplicationConfigRun.java
  6)Finish

```
package com.example;

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

@Configuration
@Profile("run")
public class ApplicationConfigRun {

   @Bean
   public ServerInfo serverInfo(){
      ServerInfo info = new ServerInfo();
      info.setIpNum("192.168.56.5");
      info.setPortNum("80");
      return info;
   }
}
```

7. MainClass 생성
  1)com.example > right-click > New > Class
  2)Name : MainClass.java

```
package com.example;
import java.util.Scanner;
```

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

    public class MainClass {
      public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Select dev or run : ");
        String config = scan.next();  //"dev" or "run"

        AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext();
        ctx.getEnvironment().setActiveProfiles(config);
        ctx.register(ApplicationConfigDev.class, ApplicationConfigRun.class);
        ctx.refresh();

        ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
        System.out.println("IP : " + info.getIpNum());
        System.out.println("Port : " + info.getPortNum());
        ctx.close();
      }
    }
```

8. 결과

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

```
      Select dev or run :
```

  2)입력시 dev를 넣으면 dev환경인 localhost/8080이 나오고, 만일 run이라고 넣으면 192.168.56.5/80이 나온다.