```
1. Environment 객체
  1)Environment 객체를 이용해서 Spring bean 설정을 할 수 있다.
    Context  ---------------------> Environment -------------------------> PropertySources
            ctx.getEnvironment()                  env.getPropertySource()    property 추가 및 추출
                                                                             추가 : propertySources.addLast()
                                                                             추출 : env.getProperty()


2. Lab
  1)In Package Explorer > right-click > New > Java Project
    -Project Name : EnvironmentDemo

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

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

    -EnvironmentDemo Project > right-click > Spring > Add Spring Project Natur

    -pom.xml file에 Spring Context Dependency 추가하기
      <version>0.0.1-SNAPSHOT</version>
      <dependencies>
        <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>4.3.24.RELEASE</version>
        </dependency>
      </dependencies>

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

  4)EnvironmentDemo/resources folder 생성
    -EnvironmentDemo project > right-click > Build Path > Configure Build Path
    -Source Tab > Add Folder
    -EnvironmentDemo click
    -Create New Folder > Folder name : resources > Finish > OK
    -EnvironmentDemo/resources(new) 확인
    -Apply and Close

  5)resources/admin.properties file 생성

    admin.id=javaexpert
    admin.pwd=12345678

  6)com.example.AdminConnection.java 생성

```

```
55      package com.example;
56
57      import org.springframework.beans.factory.DisposableBean;
58      import org.springframework.beans.factory.InitializingBean;
59      import org.springframework.context.EnvironmentAware;
60      import org.springframework.core.env.Environment;
61
62      public class AdminConnection implements EnvironmentAware, InitializingBean,
        DisposableBean{
63        private Environment env;
64        private String adminId;
65        private String adminPwd;
66
67        public void setEnv(Environment env) {
68          this.env = env;
69        }
70
71        public void setAdminId(String adminId) {
72          this.adminId = adminId;
73        }
74
75        public void setAdminPwd(String adminPwd) {
76          this.adminPwd = adminPwd;
77        }
78
79        public String getAdminId() {
80          return adminId;
81        }
82
83        public String getAdminPwd() {
84          return adminPwd;
85        }
86
87        @Override
88        public void destroy() throws Exception {
89          System.out.println("destroy()");
90        }
91
92        @Override
93        public void afterPropertiesSet() throws Exception {
94          System.out.println("afterPropertiesSet()");
95          setAdminId(env.getProperty("admin.id"));
96          setAdminPwd(env.getProperty("admin.pwd"));
97        }
98
99        //bean이 생성되기 전에 callback 으로 호출됨. 가장 먼저 호출됨.
100       //MainClass에서 사용하는 env 정보가 넘어옴.
101       @Override
102       public void setEnvironment(Environment env) {
103         System.out.println("setEnvironment()");
104         setEnv(env);
105       }
106     }
107
```

5)resources/beans.xml 생성

```
<?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>
```

6)com.example.MainClass.java 생성

```
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();
  }
}
```

7)실행

```
159      -MainClass > right-click > Run As > Java Application
160
161    8)결과
162      setEnvironment()
163      afterPropertiesSet()
164      admin ID : javaexpert
165      admin PWD : 12345678
166      ...
167      destroy()
168
169
170  3. Property file을 이용한 설정
171    1)환경에 따라 자주 변경되는 내용의 분리
172    2)XML의 Bean 설정 meta정보는 application 구조가 바뀌지 않으면 자주 변경되지 않는다.
173    3)반면에 property 값으로 제공되는 일부 설정 정보(예-DataSource Bean이 사용하는 DB 연결정보)는
       application이 동작하는 환경(개발, test, stageing, 운영)에 따라서 자주 바뀔 수 있다.
174    4)변경되는 이유와 시점이 다르다면 분리하는 것이 객체지향 설계의 기본 원칙이기에 설정에도 동일한 원칙을 적용할 수 있
       다.
175    5)환경에 따라 자주 변경될 수 있는 내용은 properties file로 분리하는 것이 가장 깔끔하다.
176    6)XML 처럼 복잡한 구성이 필요없고 키와 값의 쌍(key=value)으로 구성하면 된다.
177    7)환경에 따라 자주 변경되는 내용의 분리의 예시
178      -value속성에 설정된 값들은 환경에 따라 변경될 수 있는 내용이다.
179      -자주 변경되는 값들은 properties file에 넣어 분리하는 것이 좋다.
180
181        <beans.xml>
182        <bean id="dataSource"
183                    class="org.springframework.jdbc.datasource.SimpleDriverDataSource">
184            <property name="driverClass" value="com.mysql.jdbc.Driver" />
185            <property name="url" value="jdbc:mysql://localhost/testdb" />
186            <property name="username" value="spring" />
187            <property name="password" value="book" />
188        </bean>
189
190      -properties file로 분리한 정보는 ${}(property 치환자)을 이용하여 설정한다.
191      -${} 값을 치환해주는 기능은 <context:property-placeholder> tag에 의해 자동으로 등록되는
       PropertyPlaceHolderConfigurer Bean이 담당한다.
192
193        <database.properties>
194          db.driverClass=com.mysql.jdbc.Driver
195          db.url=jdbc:mysql://localhost/testdb
196          db.username=spring
197          db.password=book
198
199        <beans.xml>
200          <context:property-placeholder
201                location="classpath:config/database.properties" />
202          <bean id="dataSource"
203                class="org.springfremework.jdbc.datasource.SimpleDriverDataSource">
204              <property name="driverClass" value="${db.driverClass}" />
205              <property name="url" value="${db.url}" />
206              <property name="username" value="${db.username}" />
207              <property name="password" value="${db.password}" />
208          </bean>
209
```

4. Lab
  1)In Package Explorer > right-click > New > Java Project
    -Project Name : PropertyDemo

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

  3)POJO class 작성
    -com.example.Hello

```
package com.example;

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

  public Hello(){}

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

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

  public void setNames(List<String> list){
    this.names = list;
  }

  public List<String> getNames(){
    return this.names;
  }

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

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

    -com.example > right-click > New > Interface
      Interface name : Printer

```
package com.example;

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

```
    -com.example > right-click > New > Class
      Class Name : StringPrinter

        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();
          }
        }

    -com.example > right-click > New > Class
      Class Name : ConsolePrinter

        package com.example;

        public class ConsolePrinter implements Printer{

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

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

    -PropertyDemo Project > right-click > Spring > Add Spring Project Nature

    -pom.xml file에 Spring Context Dependency 추가하기
      <version>0.0.1-SNAPSHOT</version>
      <dependencies>
        <dependency>
          <groupId>org.springframework</groupId>
          <artifactId>spring-context</artifactId>
          <version>4.3.24.RELEASE</version>
        </dependency>
      </dependencies>

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

5)PropertyDemo/resources folder 생성
-PropertyDemo project > right-click > Build Path > Configure Build Path
-Source Tab > Add Folder
-PropertyDemo click
-Create New Folder > Folder name : resources > Finish > OK
-PropertyDemo/resources(new) 확인
-Apply and Close

6)Bean Configuration XML 작성
-PropertyDemo/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" />
    <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>
```

7)com.example.MainClass

```
package com.example;

import java.util.List;

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());

            List<String> list = hello.getNames();
            for(String value : list){
              System.out.println(value);
            }
          }
        }

8)실행
  -MainClass > right-click > Run As > Java Application

9)결과
  Hello Spring
  Hello Spring
  AOP
  Spring
  DI

10)jUnit으로 test
  -src/test package 생성
  -/src/test/ > right-click > New > JUnit Test Case > HelloTest > Finish
  -New JUnit Test Case창에서, Not now 선택 > OK
  -https://mvnrepository.com에서 'junit'로 검색
  -JUnit에서
  -4.12로 들어가서
  -복사해서 pom.xml로 붙여넣기
  -https://mvnrepository.com에서 'spring-test'로 검색
  -Spring TestContext Framework에서
  -4.3.24.RELEASE로 들어가서
  -복사해서 pom.xml로 붙여넣기
  -pom.xml > right-click > Run As > Maven install
    [INFO] BUILD SUCCESS 확인

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

        import java.util.List;

        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)
        @ContextConfiguration(locations="classpath:beans.xml")
        public class HelloTest {
          @Autowired
```

```
424             ApplicationContext ctx;
425
426             @Test
427             public void test() {
428               Hello hello = (Hello)ctx.getBean("hello");
429               assertEquals("Hello Spring", hello.sayHello());
430               hello.print();
431
432               Printer printer = (Printer)ctx.getBean("printer");
433               assertEquals("Hello Spring", printer.toString());
434             }
435
436             @Test
437             public void test2(){
438               Hello hello = (Hello)ctx.getBean("hello");
439
440               Hello hello2 = ctx.getBean("hello", Hello.class);
441               assertSame(hello, hello2);
442
443               assertEquals(3, hello2.getNames().size());
444             }
445           }
446
447       -right-click > Run As > Junit Test
448       -결과 -> Junit View에 초록색 bar
449
450     11)resources/value.properties 생성
451
452       myname=Spring
453       myprinter=printer
454       value1=HTML5
455       value2=CSS3
456       value3=JavaScript
457
458     12)/resources/beans.xml 에서 [Namespaces] tab
459       -목록에서 'context-http://www.springframework.org/schema/context' check
460       -<context:property-placeholder />를 사용하기 위해서
461
462         <context:property-placeholder
463               location="classpath:value.properties" />
464
465         <bean id="hello" class="com.example.Hello">
466           <property name="name" value="${myname}" />
467           <property name="printer" ref="${myprinter}" />
468           <property name="names">
469             <list>
470               <value>${value1}</value>
471               <value>${value2}</value>
472               <value>${value3}</value>
473             </list>
474           </property>
475         </bean>
476
477         <bean id="printer" class="com.example.StringPrinter" />
```

```
478         <bean id="consolePrinter" class="com.example.ConsolePrinter" />
479
480     13)Test
481       -com.example.MainClass.java
482         --right-click > Run As > Java Application
483           Hello Spring
484           Hello Spring
485           HTML5
486           CSS3
487           JavaScript
488
489       -/src/test/java/HelloTest.java
490         --right-click > Run As > JUnit Test
491         --Green Bar
492
493     14)resources/value.properties 수정
494         myname=Spring
495         myprinter=printer
496         value1=JUnit
497         value2=AOP
498         value3=DI
499         printer1=stringPrinter
500         printer2=consolePrinter
501
502     15)Hello.java code 수정
503       -com.example/Hello.java
504
505         package com.example;
506
507         import java.util.List;
508
509         import org.springframework.beans.factory.annotation.Value;
510         import org.springframework.stereotype.Component;
511         import javax.annotation.Resource;
512
513         @Component("hello")
514         public class Hello {
515           @Value("${myname}")
516           private String name;
517
518           @Resource(name="${printer1}")
519           private Printer printer;
520
521           @Value("${value1}, ${value2}, ${value3}")
522           private List<String> names;
523
524           public List<String> getNames(){
525             return names;
526           }
527           public String sayHello(){
528             return "Hello " + name;
529           }
530
531           public void print(){
```

```
532             this.printer.print(sayHello());
533           }
534         }
535
536     16)StringPrinter.java 수정
537         package com.example;
538
539         import org.springframework.stereotype.Component;
540
541         @Component("stringPrinter")
542         public class StringPrinter implements Printer {
543           private StringBuffer buffer =  new StringBuffer();
544
545           @Override
546           public void print(String message) {
547             this.buffer.append(message);
548           }
549
550           @Override
551           public String toString(){
552             return this.buffer.toString();
553           }
554         }
555
556     17)beans.xml 수정하기
557
558         <?xml version="1.0" encoding="UTF-8"?>
559         <beans xmlns="http://www.springframework.org/schema/beans"
560           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
561           xmlns:context="http://www.springframework.org/schema/context"
562           xsi:schemaLocation="http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans.xsd
563               http://www.springframework.org/schema/context
                  http://www.springframework.org/schema/context/spring-context-3.2.xsd">
564
565           <context:property-placeholder location="classpath:value.properties "/>
566           <context:component-scan base-package="com.example" />
567         </beans>
568
569     18)MainClass.java 수정하기
570
571         package com.example;
572
573         import java.util.List;
574
575         import org.springframework.context.ApplicationContext;
576         import org.springframework.context.support.GenericXmlApplicationContext;
577
578         public class MainClass {
579           public static void main(String [] args){
580             ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");
581
582             Hello hello = (Hello)ctx.getBean("hello");
583             System.out.println(hello.sayHello());
```

```
            hello.print();

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

            List<String> list = hello.getNames();
            for(String value : list){
              System.out.println(value);
            }
          }
        }
```

19)실행
-MainClass > right-click > Run As > Java Application

20)결과
```
Hello Spring
Hello Spring
JUnit, AOP, DI
```

5. Lab
1)In Package Explorer > right-click > New > Java Project
-Project Name : PropertyDemo1

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

3)/src/com.example.AdminConnection.java 생성
```
package com.example;

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

  public String getAdminId() {
    return adminId;
  }

  public void setAdminId(String adminId) {
    this.adminId = adminId;
  }

  public String getAdminPwd() {
    return adminPwd;
  }

  public void setAdminPwd(String adminPwd) {
    this.adminPwd = adminPwd;
  }

  public String getSubAdminId() {
```

```
638         return subAdminId;
639       }
640
641       public void setSubAdminId(String subAdminId) {
642         this.subAdminId = subAdminId;
643       }
644
645       public String getSubAdminPwd() {
646         return subAdminPwd;
647       }
648
649       public void setSubAdminPwd(String subAdminPwd) {
650         this.subAdminPwd = subAdminPwd;
651       }
652     }
653
654   4)Java Project를 Spring Project로 변환
655     -PropertyDemo1 Project > right-click > Configuration > Convert to Maven Project
656       --Project : /PropertyDemo1
657       --Group Id : PropertyDemo1
658       --Artifact Id : PropertyDemo1
659       --version : 0.0.1-SNAPSHOT
660       --Packaging : jar
661       --Finish
662
663     -PropertyDemo1 Project > right-click > Spring > Add Spring Project Nature
664
665     -pom.xml file에 Spring Context Dependency 추가하기
666       <version>0.0.1-SNAPSHOT</version>
667       <dependencies> <--- dependencies element 추가
668         <dependency> <---여기에 paste
669           <groupId>org.springframework</groupId>
670           <artifactId>spring-context</artifactId>
671           <version>4.3.24.RELEASE</version>
672         </dependency>
673       </dependencies>
674
675     -pom.xml > right-click > Run As > Maven install
676       [INFO] BUILD SUCCESS 확인
677
678   5)PropertyDemo/resources folder 생성
679     -PropertyDemo1 project > right-click > Build Path > Configure Build Path
680     -Source Tab > Add Folder
681     -PropertyDemo1 click
682     -Create New Folder > Folder name : resources > Finish > OK
683     -PropertyDemo1/resources(new) 확인
684     -Apply and Close
685
686   6)  /resources 두 개의 properties file 생성
687
688     <admin.properties>
689       admin.id=javaexpert
690       admin.pwd=12345678
691
```

```
692     <sub.admin.properties>
693       sub.admin.id=javasoft
694       sub.admin.pwd=987654321
695
696   7)Bean Configuration XML 작성
697     -PropertyDemo1/resources > right-click > New > Other > Spring > Spring Bean
        Configuration File
698     -File name : beans.xml > Finish
699     -Namespace tab에서 context - http://www.springframework.org/schema/context check
700
701       <?xml version="1.0" encoding="UTF-8"?>
702       <beans xmlns="http://www.springframework.org/schema/beans"
703         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
704         xmlns:context="http://www.springframework.org/schema/context"
705         xsi:schemaLocation="http://www.springframework.org/schema/beans
          http://www.springframework.org/schema/beans/spring-beans.xsd
706           http://www.springframework.org/schema/context
              http://www.springframework.org/schema/context/spring-context-3.2.xsd">
707
708         <context:property-placeholder location="classpath:admin.properties,
          classpath:sub.admin.properties" />
709
710         <bean id="adminConnection" class="com.example.AdminConnection">
711           <property name="adminId">
712             <value>${admin.id}</value>
713           </property>
714           <property name="adminPwd">
715             <value>${admin.pwd}</value>
716           </property>
717           <property name="subAdminId">
718             <value>${sub.admin.id}</value>
719           </property>
720           <property name="subAdminPwd">
721             <value>${sub.admin.pwd}</value>
722           </property>
723         </bean>
724       </beans>
725
726   8)/src/com.example.MainClass.java 생성
727     package com.example;
728
729     import org.springframework.context.support.AbstractApplicationContext;
730     import org.springframework.context.support.GenericXmlApplicationContext;
731
732     public class MainClass {
733       public static void main(String [] args){
734         AbstractApplicationContext ctx =
735               new GenericXmlApplicationContext("classpath:beans.xml");
736         AdminConnection connection = ctx.getBean("adminConnection",
          AdminConnection.class);
737         System.out.println("admin ID : " + connection.getAdminId());
738         System.out.println("admin PWD : " + connection.getAdminPwd());
739         System.out.println("sub admin ID : " + connection.getSubAdminId());
740         System.out.println("sub admin PWD : " + connection.getSubAdminPwd());
```

```
741
742           ctx.close();
743         }
744       }
745
746     9)결과
747       -MainClass.java > right-click > Run As > Java Application
748         admin ID : javaexpert
749         admin PWD : 12345678
750         sub admin ID : javasoft
751         sub admin PWD : 987654321
752
753
754   6. Lab
755     1)In Package Explorer > right-click > New > Java Project
756       -Project Name : PropertyDemo2
757
758     2)/src/ right-click > New > Package
759       -Package name : com.example
760
761     3)/src/com.example.AdminConnection.java 생성
762       package com.example;
763
764       public class AdminConnection {
765         private String adminId;
766         private String adminPwd;
767         private String subAdminId;
768         private String subAdminPwd;
769
770         public String getAdminId() {
771           return adminId;
772         }
773
774         public void setAdminId(String adminId) {
775           this.adminId = adminId;
776         }
777
778         public String getAdminPwd() {
779           return adminPwd;
780         }
781
782         public void setAdminPwd(String adminPwd) {
783           this.adminPwd = adminPwd;
784         }
785
786         public String getSubAdminId() {
787           return subAdminId;
788         }
789
790         public void setSubAdminId(String subAdminId) {
791           this.subAdminId = subAdminId;
792         }
793
794         public String getSubAdminPwd() {
```

```
795         return subAdminPwd;
796       }
797
798       public void setSubAdminPwd(String subAdminPwd) {
799         this.subAdminPwd = subAdminPwd;
800       }
801     }
802
803   4)Java Project를 Spring Project로 변환
804     -PropertyDemo1 Project > right-click > Configuration > Convert to Maven Project
805       --Project : /PropertyDemo2
806       --Group Id : PropertyDemo2
807       --Artifact Id : PropertyDemo2
808       --version : 0.0.1-SNAPSHOT
809       --Packaging : jar
810       --Finish
811
812     -PropertyDemo2 Project > right-click > Spring > Add Spring Project Nature
813
814     -pom.xml 파일에 Spring Context Dependency 추가하기
815       <version>0.0.1-SNAPSHOT</version>
816       <dependencies>
817         <dependency>
818           <groupId>org.springframework</groupId>
819           <artifactId>spring-context</artifactId>
820           <version>4.3.24.RELEASE</version>
821         </dependency>
822       </dependencies>
823
824     -pom.xml > right-click > Run As > Maven install
825       [INFO] BUILD SUCCESS 확인
826
827   5)PropertyDemo/resources folder 생성
828     -PropertyDemo2 project > right-click > Build Path > Configure Build Path
829     -Source Tab > Add Folder
830     -PropertyDemo2 click
831     -Create New Folder > Folder name : resources > Finish > OK
832     -PropertyDemo2/resources(new) 확인
833     -Apply and Close
834
835   6)  /resources 두 개의 properties file 생성
836
837     <admin.properties>
838       admin.id=javaexpert
839       admin.pwd=12345678
840
841     <sub.admin.properties>
842       sub.admin.id=javasoft
843       sub.admin.pwd=987654321
844
845   7)/src/com.example.ApplicationConfig.java>
846       package com.example;
847
848       import org.springframework.beans.factory.annotation.Value;
```

```
849         import org.springframework.context.annotation.Bean;
850         import org.springframework.context.annotation.Configuration;
851         import org.springframework.context.support.PropertySourcesPlaceholderConfigurer;
852         import org.springframework.core.io.ClassPathResource;
853         import org.springframework.core.io.Resource;
854
855         @Configuration
856         public class ApplicationConfig {
857           @Value("${admin.id}")
858           private String adminId;
859           @Value("${admin.pwd}")
860           private String adminPwd;
861           @Value("${sub.admin.id}")
862           private String subAdminId;
863           @Value("${sub.admin.pwd}")
864           private String subAdminPwd;
865
866           @Bean
867           public static PropertySourcesPlaceholderConfigurer Properties(){
868             PropertySourcesPlaceholderConfigurer configurer =
869                  new PropertySourcesPlaceholderConfigurer();
870
871             Resource [] locations = new Resource[2];
872             locations[0] = new ClassPathResource("admin.properties");
873             locations[1] = new ClassPathResource("sub.admin.properties");
874             configurer.setLocations(locations);
875
876             return configurer;
877           }
878
879         @Bean
880         public AdminConnection adminConfig(){
881           AdminConnection adminConnection = new AdminConnection();
882           adminConnection.setAdminId(adminId);
883           adminConnection.setAdminPwd(adminPwd);
884           adminConnection.setSubAdminId(subAdminId);
885           adminConnection.setSubAdminPwd(subAdminPwd);
886           return adminConnection;
887         }
888       }
889
890     8)/src/com.example.MainClass.java
891         package com.example;
892
893         import org.springframework.context.annotation.AnnotationConfigApplicationContext;
894
895         public class MainClass {
896           public static void main(String[] args) {
897             AnnotationConfigApplicationContext ctx =
898                  new AnnotationConfigApplicationContext(ApplicationConfig.class);
899             AdminConnection conn = ctx.getBean("adminConfig", AdminConnection.class);
900
901             System.out.println("admin ID : " + conn.getAdminId());
902             System.out.println("admin PWD : " + conn.getAdminPwd());
```

```
            System.out.println("sub admin ID : " + conn.getSubAdminId());
            System.out.println("sub admin PWD : " + conn.getSubAdminPwd());
          }
        }
```

9)결과
-MainClass.java > right-click > Run As > Java Application
admin ID : javaexpert
admin PWD : 12345678
sub admin ID : javasoft
sub admin PWD : 987654321

7. Profile 속성을 이용한 설정
-동일한 Spring Bean을 여러 개 만들어 놓고 상황(환경)에 따라서 적절한 Spring bean을 사용할 수 있다.
-profile 속성을 사용한다.
-역시 Java file을 이용하는 방법과 XML 설정 file을 이용하는 방법이 있다.

8. Lab
1)In Package Explorer > right-click > New > Java Project
-Project Name : ProfileDemo

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

3)Java Project를 Spring Project로 변환
-ProfileDemo Project > right-click > Configuration > Convert to Maven Project
--Project : /ProfileDemo
--Group Id : ProfileDemo
--Artifact Id : ProfileDemo
--version : 0.0.1-SNAPSHOT
--Packaging : jar
--Finish
-ProfileDemo Project > right-click > Spring > Add Spring Project Nature

-pom.xml file에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>4.3.24.RELEASE</version>
  </dependency>
</dependencies>
```

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

4)ProfileDemo/resources folder 생성
-ProfileDemo project > right-click > Build Path > Configure Build Path
-Source Tab > Add Folder
-ProfileDemo click

```
   -Create New Folder > Folder name : resources > Finish > OK
   -ProfileDemo/resources(new) 확인
   -Apply and Close

5)ServerInfo.java 생성
   -/src/com.example.ServerInfo.java

     package com.example;

     public class ServerInfo {
       private String ipNum;
       private String portNum;
       public String getIpNum() {
         return ipNum;
       }
       public void setIpNum(String ipNum) {
         this.ipNum = ipNum;
       }
       public String getPortNum() {
         return portNum;
       }
       public void setPortNum(String portNum) {
         this.portNum = portNum;
       }
     }

6)  XML 설정 file 2개 생성
   -/resource > right-click > New > Spring Bean Configuration File
   -File name : run.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"
         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>

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

         <bean id="serverInfo" class="com.example.ServerInfo">
```

```
            <property name="ipNum" value="localhost" />
            <property name="portNum" value="8080" />
          </bean>
        </beans>

  7)MainClass 생성
    -/src/com.example.MainClass.java

      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();
        }
      }

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


9. Lab
  1)In Package Explorer > right-click > New > Java Project
    -Project Name : ProfileDemo1

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

  3)Java Project를 Spring Project로 변환
    -ProfileDemo1 Project > right-click > Configuration > Convert to Maven Project
      --Project : /ProfileDemo1
      --Group Id : ProfileDemo1
      --Artifact Id : ProfileDemo1
      --version : 0.0.1-SNAPSHOT
      --Packaging : jar
      --Finish
```

-ProfileDemo1 Project > right-click > Spring > Add Spring Project Nature

-pom.xml file에 Spring Context Dependency 추가하기

```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>4.3.24.RELEASE</version>
  </dependency>
</dependencies>
```

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

[INFO] BUILD SUCCESS 확인

4)ServerInfo.java 생성

-/src/com.example.ServerInfo.java

```
package com.example;

public class ServerInfo {
  private String ipNum;
  private String portNum;
  public String getIpNum() {
    return ipNum;
  }
  public void setIpNum(String ipNum) {
    this.ipNum = ipNum;
  }
  public String getPortNum() {
    return portNum;
  }
  public void setPortNum(String portNum) {
    this.portNum = portNum;
  }
}
```

5)Java 설정 file 2개 생성

-/src/com.example.ApplicationConfigDev.java

```
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;
          }
        }

    -/src/com.example.ApplicationConfigRun.java

        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;
          }
        }

    6)MainClass 생성
      -/src/com.example.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();
          }
        }

    7)결과
```

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