1. Environment 객체
  1)Environment 객체를 이용해서 스프링 빈 설정을 할 수 있다.
    Context ------------------------> Environment ----------------------------> PropertySources
            ctx.getEnvironment()   env.getPropertySource()   property 추가 및 추출
                                                                  추가 : propertySources.addLast()
                                                                  추출 : env.getPro
perty()

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
      --Package Explorer에서 보이는 Project 아이콘에 Maven의 'M'자가 보임.

    -EnvironmentDemo Project > right-click > Spring > Add Spring Project Natur
      --Package Explorer에서 보이는 Project 아이콘에 'M'자와 Spring의 'S'가 보임.

    -pom.xml 파일에 Spring Context Dependency 추가하기
      <version>0.0.1-SNAPSHOT</version>
      <dependencies> <--- dependencies element 추가
        <dependency> <---여기에 paste
        <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 파일 생성

    admin.id=javaexpert

```
53      admin.pwd=12345678
54
55    6)com.example.AdminConnection.java 생성
56
57      package com.example;
58
59      import org.springframework.beans.factory.DisposableBean;
60      import org.springframework.beans.factory.InitializingBean;
61      import org.springframework.context.EnvironmentAware;
62      import org.springframework.core.env.Environment;
63
64      public class AdminConnection implements EnvironmentAware, InitializingBean,
        DisposableBean{
65        private Environment env;
66        private String adminId;
67        private String adminPwd;
68
69        public void setEnv(Environment env) {
70          this.env = env;
71        }
72
73        public void setAdminId(String adminId) {
74          this.adminId = adminId;
75        }
76
77        public void setAdminPwd(String adminPwd) {
78          this.adminPwd = adminPwd;
79        }
80
81        public String getAdminId() {
82          return adminId;
83        }
84
85        public String getAdminPwd() {
86          return adminPwd;
87        }
88
89        @Override
90        public void destroy() throws Exception {
91          System.out.println("destroy()");
92        }
93
94        @Override
95        public void afterPropertiesSet() throws Exception {
96          System.out.println("afterPropertiesSet()");
97          setAdminId(env.getProperty("admin.id"));
98          setAdminPwd(env.getProperty("admin.pwd"));
99        }
100
101       //bean이 생성되기 전에 callback 으로 호출됨. 가장 먼저 호출됨.
102       //MainClass에서 사용하는 env 정보가 넘어옴.
103       @Override
104       public void setEnvironment(Environment env) {
105         System.out.println("setEnvironment()");
```

```
106         setEnv(env);
107       }
108     }
109
110   5)resources/beans.xml 생성
111
112     <?xml version="1.0" encoding="UTF-8"?>
113     <beans xmlns="http://www.springframework.org/schema/beans"
114       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
115       xsi:schemaLocation="http://www.springframework.org/schema/beans
          http://www.springframework.org/schema/beans/spring-beans.xsd">
116
117       <bean id="adminConnection" class="com.example.AdminConnection" />
118
119     </beans>
120
121   6)com.example.MainClass.java 생성
122
123     package com.example;
124
125     import java.io.IOException;
126
127     import org.springframework.context.ConfigurableApplicationContext;
128     import org.springframework.context.support.GenericXmlApplicationContext;
129     import org.springframework.core.env.ConfigurableEnvironment;
130     import org.springframework.core.env.MutablePropertySources;
131     import org.springframework.core.io.support.ResourcePropertySource;
132
133     public class MainClass {
134       public static void main(String [] args){
135         ConfigurableApplicationContext ctx = new GenericXmlApplicationContext();
136         ConfigurableEnvironment env = ctx.getEnvironment();
137
138         MutablePropertySources propertySouces = env.getPropertySources();
139         //내가 원하는 정보를 얻을 때까지 모든 propertySources를 앞에서 부터 차례로 모두 검색함.
140         try{
141           propertySouces.addLast(new ResourcePropertySource("classpath:admin.properties"));
              //property 추가
142
143           System.out.println(env.getProperty("admin.id"));    //property 추출
144           System.out.println(env.getProperty("admin.pwd"));
145         }catch(IOException ex){}
146
147         GenericXmlApplicationContext gCtx = (GenericXmlApplicationContext)ctx;
148         gCtx.load("classpath:beans.xml");
149         gCtx.refresh();
150
151         AdminConnection adminConnection = gCtx.getBean("adminConnection",
          AdminConnection.class);
152         System.out.println("admin ID : " + adminConnection.getAdminId());
153         System.out.println("admin PWD : " + adminConnection.getAdminPwd());
154
155         gCtx.close();
156         ctx.close();
```

```
      }
    }

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

  8)결과
    setEnvironment()
    afterPropertiesSet()
    admin ID : javaexpert
    admin PWD : 12345678
    ...
    destroy()



3. Property 파일을 이용한 설정
  1)환경에 따라 자주 변경되는 내용의 분리
  2)XML의 Bean 설정 메타정보는 어플리케이션 구조가 바뀌지 않으면 자주 변경되지 않는다.
  3)반면에 프로퍼티 값으로 제공되는 일부 설정정보(예-DataSource Bean이 사용하는 DB 연결정보)는 어플리케이션이
  동작하는 환경(개발, 테스트, 스테이징, 운영)에 따라서 자주 바뀔 수 있다.
  4)변경되는 이유와 시점이 다르다면 분리하는 것이 객체지향 설계의 기본 원칙이기에 설정에도 동일한 원칙을 적용할 수 있다.
  5)환경에 따라 자주 변경될 수 있는 내용은 properties 파일로 분리하는 것이 가장 깔끔하다.
  6)XML 처럼 복잡한 구성이 필요없고 키와 값의 쌍(key=value)으로 구성하면 된다.
  7)환경에 따라 자주 변경되는 내용의 분리의 예시
    -value속성에 설정된 값들은 환경에 따라 변경될 수 있는 내용이다.
    -자주 변경되는 값들은 properties 파일에 넣어 분리하는 것이 좋다.

      <beans.xml>
      <bean id="dataSource"
                class="org.springframework.jdbc.datasource.SimpleDriverDataSource">
          <property name="driverClass" value="com.mysql.jdbc.Driver" />
          <property name="url" value="jdbc:mysql://localhost/testdb" />
          <property name="username" value="spring" />
          <property name="password" value="book" />
      </bean>

    -properties 파일로 분리한 정보는 ${}(property 치환자)을 이용하여 설정한다.
    -${} 값을 치환해주는 기능은 <context:property-placeholder> 태그에 의해 자동으로 등록되는
    PropertyPlaceHolderConfigurer Bean이 담당한다.

      <database.properties>
        db.driverClass=com.mysql.jdbc.Driver
        db.url=jdbc:mysql://localhost/testdb
        db.username=spring
        db.password=book

      <beans.xml>
        <context:property-placeholder
              location="classpath:config/database.properties" />
        <bean id="dataSource"
              class="org.springfremework.jdbc.datasource.SimpleDriverDataSource">
            <property name="driverClass" value="${db.driverClass}" />
            <property name="url" value="${db.url}" />
```

```
            <property name="username" value="${db.username}" />
            <property name="password" value="${db.password}" />
        </bean>


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;

```

```
263         public interface Printer{
264           void print(String message);
265         }
266
267     -com.example > right-click > New > Class
268       Class Name : StringPrinter
269
270         package com.example;
271
272         public class StringPrinter implements Printer{
273           private StringBuffer buffer = new StringBuffer();
274
275           @Override
276           public void print(String message){
277             this.buffer.append(message);
278           }
279
280           public String toString(){
281             return this.buffer.toString();
282           }
283         }
284
285     -com.example > right-click > New > Class
286       Class Name : ConsolePrinter
287
288         package com.example;
289
290         public class ConsolePrinter implements Printer{
291
292           @Override
293           public void print(String message){
294             System.out.println(message);
295           }
296         }
297
298   4)Java Project를 Spring Project로 변환
299     -PropertyDemo Project > right-click > Configuration > Convert to Maven Project
300       --Project : /PropertyDemo
301       --Group Id : PropertyDemo
302       --Artifact Id : PropertyDemo
303       --version : 0.0.1-SNAPSHOT
304       --Packaging : jar
305       --Finish
306       --Package Explorer에서 보이는 Project 아이콘에 Maven의 'M'자가 보임.
307
308     -PropertyDemo Project > right-click > Spring > Add Spring Project Nature
309       --Package Explorer에서 보이는 Project 아이콘에 'M'자와 Spring의 'S'가 보임.
310
311     -pom.xml 파일에 Spring Context Dependency 추가하기
312       <version>0.0.1-SNAPSHOT</version>
313       <dependencies> <--- dependencies element 추가
314         <dependency> <---여기에 paste
315           <groupId>org.springframework</groupId>
316           <artifactId>spring-context</artifactId>
```

```
317             <version>4.3.24.RELEASE</version>
318           </dependency>
319         </dependencies>
320
321       -pom.xml > right-click > Run As > Maven install
322         [INFO] BUILD SUCCESS 확인
323
324     5)PropertyDemo/resources folder 생성
325       -PropertyDemo project > right-click > Build Path > Configure Build Path
326       -Source Tab > Add Folder
327       -PropertyDemo click
328       -Create New Folder > Folder name : resources > Finish > OK
329       -PropertyDemo/resources(new) 확인
330       -Apply and Close
331
332     6)Bean Configuration XML 작성
333       -PropertyDemo/resources > right-click > New > Other > Spring > Spring Bean Configuration
          File
334       -File name : beans.xml > Finish
335
336         <?xml version="1.0" encoding="UTF-8"?>
337         <beans xmlns="http://www.springframework.org/schema/beans"
338           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
339           xsi:schemaLocation="http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans.xsd">
340
341           <bean id="hello" class="com.example.Hello">
342             <property name="name" value="Spring" />
343             <property name="printer" ref="printer" />
344             <property name="names">
345               <list>
346                 <value>AOP</value>
347                 <value>Spring</value>
348                 <value>DI</value>
349               </list>
350             </property>
351           </bean>
352
353           <bean id="printer" class="com.example.StringPrinter" />
354           <bean id="consolePrinter" class="com.example.ConsolePrinter" />
355
356         </beans>
357
358     7)com.example.MainClass
359
360         package com.example;
361
362         import java.util.List;
363
364         import org.springframework.context.ApplicationContext;
365         import org.springframework.context.support.GenericXmlApplicationContext;
366
367         public class MainClass {
368           public static void main(String [] args){
```

```
369             ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");
370
371             Hello hello = (Hello)ctx.getBean("hello");
372             System.out.println(hello.sayHello());
373             hello.print();
374
375             Printer printer = ctx.getBean("printer", StringPrinter.class);
376             System.out.println(printer.toString());
377
378             List<String> list = hello.getNames();
379             for(String value : list){
380               System.out.println(value);
381             }
382           }
383         }
384
385     8)실행
386       -MainClass > right-click > Run As > Java Application
387
388     9)결과
389       Hello Spring
390       Hello Spring
391       AOP
392       Spring
393       DI
394
395     10)jUnit으로 테스트
396       -src/test package 생성
397       -/src/test/ > right-click > New > JUnit Test Case > HelloTest > Finish
398       -New JUnit Test Case창에서, Not now 선택 > OK
399       -https://mvnrepository.com에서 'junit'로 검색
400       -JUnit에서
401       -4.12로 들어가서
402       -복사해서 pom.xml로 붙여넣기
403       -https://mvnrepository.com에서 'spring-test'로 검색
404       -Spring TestContext Framework에서
405       -4.3.24.RELEASE로 들어가서
406       -복사해서 pom.xml로 붙여넣기
407       -pom.xml > right-click > Run As > Maven install
408         [INFO] BUILD SUCCESS 확인
409
410           import static org.junit.Assert.assertEquals;
411           import static org.junit.Assert.assertSame;
412
413           import java.util.List;
414
415           import org.junit.Test;
416           import org.junit.runner.RunWith;
417           import org.springframework.beans.factory.annotation.Autowired;
418           import org.springframework.context.ApplicationContext;
419           import org.springframework.test.context.ContextConfiguration;
420           import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
421
422           import com.example.Hello;
```

```
import com.example.Printer;

@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations="classpath:beans.xml")
public class HelloTest {
  @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());
  }
}
```

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

11)resources/value.properties 생성

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

12)/resources/beans.xml 에서 [Namespaces] tab
-목록에서 'context-http://www.springframework.org/schema/context' check
-<context:property-placeholder />를 사용하기 위해서

```
<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" />

13)Test
  -com.example.MainClass.java
    --right-click > Run As > Java Application
      Hello Spring
      Hello Spring
      HTML5
      CSS3
      JavaScript

  -/src/test/java/HelloTest.java
    --right-click > Run As > JUnit Test
    --Green Bar

14)resources/value.properties 수정
    myname=Spring
    myprinter=printer
    value1=JUnit
    value2=AOP
    value3=DI
    printer1=stringPrinter
    printer2=consolePrinter

15)Hello.java 코드 수정
  -com.example/Hello.java

    package com.example;

    import java.util.List;

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

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

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

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

      public List<String> getNames(){
        return names;
```

```
531         }
532         public String sayHello(){
533           return "Hello " + name;
534         }
535
536         public void print(){
537           this.printer.print(sayHello());
538         }
539       }
540
541   16)StringPrinter.java 수정
542       package com.example;
543
544       import org.springframework.stereotype.Component;
545
546       @Component("stringPrinter")
547       public class StringPrinter implements Printer {
548         private StringBuffer buffer =  new StringBuffer();
549
550         @Override
551         public void print(String message) {
552           this.buffer.append(message);
553         }
554
555         @Override
556         public String toString(){
557           return this.buffer.toString();
558         }
559       }
560
561   17)beans.xml 수정하기
562
563       <?xml version="1.0" encoding="UTF-8"?>
564       <beans xmlns="http://www.springframework.org/schema/beans"
565         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
566         xmlns:context="http://www.springframework.org/schema/context"
567         xsi:schemaLocation="http://www.springframework.org/schema/beans
          http://www.springframework.org/schema/beans/spring-beans.xsd
568           http://www.springframework.org/schema/context
            http://www.springframework.org/schema/context/spring-context-3.2.xsd">
569
570         <context:property-placeholder location="classpath:value.properties "/>
571         <context:component-scan base-package="com.example" />
572       </beans>
573
574   18)MainClass.java 수정하기
575
576       package com.example;
577
578       import java.util.List;
579
580       import org.springframework.context.ApplicationContext;
581       import org.springframework.context.support.GenericXmlApplicationContext;
582
```

```
583         public class MainClass {
584           public static void main(String [] args){
585             ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");
586
587             Hello hello = (Hello)ctx.getBean("hello");
588             System.out.println(hello.sayHello());
589             hello.print();
590
591             Printer printer = ctx.getBean("stringPrinter", StringPrinter.class);
592             System.out.println(printer.toString());
593
594             List<String> list = hello.getNames();
595             for(String value : list){
596               System.out.println(value);
597             }
598           }
599         }
600
601     19)실행
602       -MainClass > right-click > Run As > Java Application
603
604     20)결과
605       Hello Spring
606       Hello Spring
607       JUnit, AOP, DI
608
609
610   5. Lab
611       ApplicationContext.xml  --> XML 파일을 이용하는 방법
612       -Spring 설정 XML 파일에 property 파일에 대해 명시한다.
613       -admin.properties
614       -sub_admin.properties
615
616       ApplicationConfig --> Java 파일을 이용하는 방법
617       -Spring 설정 Java 파일에 property 파일을 명시한다.
618       -admin.properties
619       -sub_admin.properties
620
621     1)/src/ right-click > New > Package
622       -Package name : com.example
623
624     2)/src/com.example.AdminConnection.java 생성
625       package com.example;
626
627       public class AdminConnection {
628         private String adminId;
629         private String adminPwd;
630         private String subAdminId;
631         private String subAdminPwd;
632
633         public String getAdminId() {
634           return adminId;
635         }
636
```

```
637         public void setAdminId(String adminId) {
638           this.adminId = adminId;
639         }
640
641         public String getAdminPwd() {
642           return adminPwd;
643         }
644
645         public void setAdminPwd(String adminPwd) {
646           this.adminPwd = adminPwd;
647         }
648
649         public String getSubAdminId() {
650           return subAdminId;
651         }
652
653         public void setSubAdminId(String subAdminId) {
654           this.subAdminId = subAdminId;
655         }
656
657         public String getSubAdminPwd() {
658           return subAdminPwd;
659         }
660
661         public void setSubAdminPwd(String subAdminPwd) {
662           this.subAdminPwd = subAdminPwd;
663         }
664       }
665
666     3)/resources 두 개의 properties 파일 생성
667
668       <admin.properties>
669         admin.id=javaexpert
670         admin.pwd=12345678
671
672       <sub.admin.properties>
673         sub.admin.id=javasoft
674         sub.admin.pwd=987654321
675
676     4)/resources/beans.xml 생성
677
678       <beans.xml>
679         <?xml version="1.0" encoding="UTF-8"?>
680         <beans xmlns="http://www.springframework.org/schema/beans"
681           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
682           xmlns:context="http://www.springframework.org/schema/context"
683           xsi:schemaLocation="http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans.xsd
684             http://www.springframework.org/schema/context
                http://www.springframework.org/schema/context/spring-context-3.2.xsd">
685
686           <context:property-placeholder location="classpath:admin.properties,
              classpath:sub.admin.properties" />
687
```

```
        <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>

  5)/src/com.example.MainClass.java 생성
    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();
      }
    }


6. Lab
  1)/resources 두 개의 properties 파일 생성

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

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

  2)/src/com.example.ApplicationConfig.java>

    <ApplicationConfig.java>
      package com.example;

      import org.springframework.beans.factory.annotation.Value;
```

```
742       import org.springframework.context.annotation.Bean;
743       import org.springframework.context.annotation.Configuration;
744       import org.springframework.context.support.PropertySourcesPlaceholderConfigurer;
745       import org.springframework.core.io.ClassPathResource;
746       import org.springframework.core.io.Resource;
747
748       @Configuration
749       public class ApplicationConfig {
750         @Value("${admin.id}")
751         private String adminId;
752         @Value("${admin.pwd}")
753         private String adminPwd;
754         @Value("${sub.admin.id}")
755         private String subAdminId;
756         @Value("${sub.admin.pwd}")
757         private String subAdminPwd;
758
759         @Bean
760         public static PropertySourcesPlaceholderConfigurer Properties(){
761           PropertySourcesPlaceholderConfigurer configurer =
762                 new PropertySourcesPlaceholderConfigurer();
763
764           Resource [] locations = new Resource[2];
765           locations[0] = new ClassPathResource("admin.properties");
766           locations[1] = new ClassPathResource("sub.admin.properties");
767           configurer.setLocations(locations);
768
769           return configurer;
770         }
771
772       @Bean
773       public AdminConnection adminConfig(){
774         AdminConnection adminConnection = new AdminConnection();
775         adminConnection.setAdminId(adminId);
776         adminConnection.setAdminPwd(adminPwd);
777         adminConnection.setSubAdminId(subAdminId);
778         adminConnection.setSubAdminPwd(subAdminPwd);
779         return adminConnection;
780       }
781     }
782
783   3)/src/com.example.MainClass1.java
784
785     <MainClass1.java>
786       package com.example;
787
788       import org.springframework.context.annotation.AnnotationConfigApplicationContext;
789
790       public class MainClass1 {
791         public static void main(String[] args) {
792           AnnotationConfigApplicationContext ctx =
793                 new AnnotationConfigApplicationContext(ApplicationConfig.class);
794           AdminConnection conn = ctx.getBean("adminConfig", AdminConnection.class);
795
```

```
796             System.out.println("admin ID : " + conn.getAdminId());
797             System.out.println("admin PWD : " + conn.getAdminPwd());
798             System.out.println("sub admin ID : " + conn.getSubAdminId());
799             System.out.println("sub admin PWD : " + conn.getSubAdminPwd());
800           }
801         }
802
803
804 7. Profile 속성을 이용한 설정
805   -동일한 Spring Bean을 여러개 만들어 놓고 상황(환경)에 따라서 적절한 스프링 빈을 사용할 수 있다.
806   -profile 속성을 사용한다.
807   -역시 Java 파일을 이용하는 방법과 XML 설정 파일을 이용하는 방법이 있다.
808
809
810 8. Lab
811   1)Package 생성
812     -/src/ > right-click > New > Package
813     -Package name : com.example
814
815   2)XML 설정 파일 2개 생성
816     -/resource > right-click > New > Spring Bean Configuration File
817     -File name : run.xml
818
819         <?xml version="1.0" encoding="UTF-8"?>
820         <beans xmlns="http://www.springframework.org/schema/beans"
821           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
822           xsi:schemaLocation="http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans.xsd"
823           profile="run">    <---이것이 핵심
824
825           <bean id="serverInfo" class="com.example.ServerInfo">
826             <property name="ipNum" value="192.168.56.5" />
827             <property name="portNum" value="80" />
828           </bean>
829         </beans>
830
831     -/resource > right-click > New > Spring Bean Configuration File
832     -File name : dev.xml
833
834         <?xml version="1.0" encoding="UTF-8"?>
835         <beans xmlns="http://www.springframework.org/schema/beans"
836           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
837           xsi:schemaLocation="http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans.xsd"
838           profile="dev">    <---이것이 핵심
839
840           <bean id="serverInfo" class="com.example.ServerInfo">
841             <property name="ipNum" value="192.168.56.5" />
842             <property name="portNum" value="80" />
843           </bean>
844         </beans>
845
846   3)ServerInfo.java 생성
847     -/src/com.example.ServerInfo.java
```

```
848
849         package com.example;
850
851         public class ServerInfo {
852           private String ipNum;
853           private String portNum;
854           public String getIpNum() {
855             return ipNum;
856           }
857           public void setIpNum(String ipNum) {
858             this.ipNum = ipNum;
859           }
860           public String getPortNum() {
861             return portNum;
862           }
863           public void setPortNum(String portNum) {
864             this.portNum = portNum;
865           }
866         }
867
868     4)MainClass 생성
869       -/src/com.example.MainClass.java
870
871         package com.example;
872         import java.util.Scanner;
873
874         import org.springframework.context.support.GenericXmlApplicationContext;
875
876         public class MainClass {
877           public static void main(String[] args) {
878             Scanner scan = new Scanner(System.in);
879             System.out.print("Select dev or run : ");
880             String config = scan.next();  //"dev" or "run"
881
882             GenericXmlApplicationContext ctx = new GenericXmlApplicationContext();
883             ctx.getEnvironment().setActiveProfiles(config);
884             ctx.load("dev.xml", "run.xml");
885
886             ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
887             System.out.println("IP : " + info.getIpNum());
888             System.out.println("Port : " + info.getPortNum());
889             ctx.close();
890           }
891         }
892
893     5)결과
894       -입력시 dev를 넣으면 dev환경인 localhost/8080이 나오고, 만일 run이라고 넣으면 192.168.56.5/80이 나온다.
895
896
897 9. Lab
898     1)Package 생성
899       -/src/ > right-click > New > Package
900       -Package name : com.example
901
```

2)Java 설정 파일 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;
  }
}
```

3)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;
  }
```

```
956         public void setIpNum(String ipNum) {
957           this.ipNum = ipNum;
958         }
959         public String getPortNum() {
960           return portNum;
961         }
962         public void setPortNum(String portNum) {
963           this.portNum = portNum;
964         }
965       }
966
967   4)MainClass 생성
968     -/src/com.example.MainClass.java
969
970       package com.example;
971       import java.util.Scanner;
972
973       import org.springframework.context.annotation.AnnotationConfigApplicationContext;
974
975       public class MainClass {
976         public static void main(String[] args) {
977           Scanner scan = new Scanner(System.in);
978           System.out.print("Select dev or run : ");
979           String config = scan.next();  //"dev" or "run"
980
981           AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext();
982           ctx.getEnvironment().setActiveProfiles(config);
983           ctx.register(ApplicationConfigDev.class, ApplicationConfigRun.class);
984           ctx.refresh();
985
986           ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
987           System.out.println("IP : " + info.getIpNum());
988           System.out.println("Port : " + info.getPortNum());
989           ctx.close();
990         }
991       }
992
993   5)결과
994     -입력시 dev를 넣으면 dev환경인 localhost/8080이 나오고, 만일 run이라고 넣으면 192.168.56.5/80이 나온다.
```