

```

1 1. Environment 객체
2 1)Environment 객체를 이용해서 스프링 빈 설정을 할 수 있다.
3 Context -----> Environment -----> PropertySources
4         ctx.getEnvironment()           env.getPropertySource()   property 추가
5                                     및 추출
6
7                                     추가 :
8                                     propertySources.addLast()
9                                     추출 : env.getProperty()
10
11 2. Lab
12 1)New > Spring Legacy Project > Simple Projects > Simple Spring Maven
13     Project Name : EnvironmentDemo
14
15 2)/src/main/resources/admin.properties 파일 생성
16     admin.id=javaexpert
17     admin.pwd=12345678
18
19 3)/src/main/java/info.javaexpert package 생성
20
21 4)/src/main/java/info.javaexpert.AdminConnection.java 생성
22
23     package info.javaexpert;
24
25     import org.springframework.beans.factory.DisposableBean;
26     import org.springframework.beans.factory.InitializingBean;
27     import org.springframework.context.EnvironmentAware;
28     import org.springframework.core.env.Environment;
29
30     public class AdminConnection implements EnvironmentAware, InitializingBean, DisposableBean{
31         private Environment env;
32         private String adminId;
33         private String adminPwd;
34
35         public void setEnv(Environment env) {
36             this.env = env;
37         }
38
39         public void setAdminId(String adminId) {
40             this.adminId = adminId;
41         }
42
43         public void setAdminPwd(String adminPwd) {
44             this.adminPwd = adminPwd;
45         }
46
47         public String getAdminId() {
48             return adminId;
49         }
50
51         public String getAdminPwd() {
52             return adminPwd;
53         }
54
55         @Override
56         public void destroy() throws Exception {
57             System.out.println("destroy()");
58         }
59
60         @Override
61         public void afterPropertiesSet() throws Exception {
62             System.out.println("afterPropertiesSet()");
63             setAdminId(env.getProperty("admin.id"));
64             setAdminPwd(env.getProperty("admin.pwd"));
65         }

```

```

66
67 //bean이 생성되기 전에 callback 으로 호출됨. 가장 먼저 호출됨.
68 //MainClass에서 사용하는 env 정보가 넘어옴.
69 @Override
70 public void setEnvironment(Environment env) {
71     System.out.println("setEnvironment()");
72     setEnv(env);
73 }
74 }
75

```

5)/src/main/resources/beans.xml 생성

```

76
77
78 <?xml version="1.0" encoding="UTF-8"?>
79 <beans xmlns="http://www.springframework.org/schema/beans"
80     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
81     xsi:schemaLocation="http://www.springframework.org/schema/beans
82         http://www.springframework.org/schema/beans/spring-beans.xsd">
83
84     <bean id="adminConnection" class="info.javaexpert.AdminConnection" />
85
86 </beans>
87

```

6)/src/main/java/MainClass.java 생성

```

88
89 package info.javaexpert;
90
91 import java.io.IOException;
92
93 import org.springframework.context.ConfigurableApplicationContext;
94 import org.springframework.context.support.GenericXmlApplicationContext;
95 import org.springframework.core.env.ConfigurableEnvironment;
96 import org.springframework.core.env.MutablePropertySources;
97 import org.springframework.core.io.support.ResourcePropertySource;
98
99 public class MainClass {
100     public static void main(String [] args){
101         ConfigurableApplicationContext ctx = new GenericXmlApplicationContext();
102         ConfigurableEnvironment env = ctx.getEnvironment();
103
104         MutablePropertySources propertySouces = env.getPropertySources();
105         //내가 원하는 정보를 얻을 때까지 모든 propertySources를 앞에서 부터 차례로 모두 검색함.
106         try{
107             propertySouces.addLast(new ResourcePropertySource("classpath:admin.properties"));
108             //property 추가
109
110             System.out.println(env.getProperty("admin.id")); //property 추출
111             System.out.println(env.getProperty("admin.pwd"));
112         }catch(IOException ex){}
113
114         GenericXmlApplicationContext gCtx = (GenericXmlApplicationContext)ctx;
115         gCtx.load("beans.xml");
116         gCtx.refresh();
117
118         AdminConnection adminConnection = gCtx.getBean("adminConnection",
119             AdminConnection.class);
120         System.out.println("admin ID : " + adminConnection.getAdminId());
121         System.out.println("admin PWD : " + adminConnection.getAdminPwd());
122
123         gCtx.close();
124         ctx.close();
125     }
126 }
127

```

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.springframework.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)New > Spring Legacy Project > Simple Projects > Simple Spring Maven
Project Name : PropertyDemo

2)src/main/java > Create Package
-info.javaexpert

3)POJO class 작성
-info.javaexpert > right-click > New > Class
-Class name : Hello

```
<Hello.java>
package info.javaexpert;

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){
```

```

194         this.printer = printer;
195     }
196
197     public void setNames(List<String> list){
198         this.names = list;
199     }
200
201     public List<String> getNames(){
202         return this.names;
203     }
204
205     public String sayHello(){
206         return "Hello " + name;
207     }
208
209     public void print(){
210         this.printer.print(sayHello());
211     }
212 }

```

214 -info.javaexpert > right-click > New > Interface
 215 Interface name : Printer

```

217 <Printer.java>
218     package info.javaexpert;
219
220     public interface Printer{
221         void print(String message);
222     }

```

224 -info.javaexpert > right-click > New > Class
 225 Class Name : StringPrinter

```

227 <StringPrinter.java>
228     package info.javaexpert;
229
230     public class StringPrinter implements Printer{
231         private StringBuffer buffer = new StringBuffer();
232
233         public void print(String message){
234             this.buffer.append(message);
235         }
236
237         public String toString(){
238             return this.buffer.toString();
239         }
240     }

```

242 -info.javaexpert > right-click > New > Class
 243 Class Name : ConsolePrinter

```

245 <ConsolePrinter.java>
246     package info.javaexpert;
247
248     public class ConsolePrinter implements Printer{
249         public void print(String message){
250             System.out.println(message);
251         }
252     }

```

254 4) Bean Configuration XML 작성

255 -/src/main/resources > right-click > New > Spring Bean Configuration File
 256 File name : beans.xml > Finish

```

258 <?xml version="1.0" encoding="UTF-8"?>
259 <beans xmlns="http://www.springframework.org/schema/beans"
260       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="info.javaexpert.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="info.javaexpert.StringPrinter" />
<bean id="consolePrinter" class="info.javaexpert.ConsolePrinter" />

</beans>
```

5)DI Test 클래스 작성

```
-/src/info.javaexpert > right-click > New > Class
Class Name : MainClass
```

```
package info.javaexpert;

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);
        }
    }
}
```

6)Test

```
-/src/info.javaexpert.test/HelloBeanTest.java > right-click > Run As > Java Application
```

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

7)JUnit으로 테스트

```
-/src/test/java > right-click > New > JUnit Test Case > HelloTest > Finish
```

```
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 info.javaexpert.Hello;
import info.javaexpert.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

8)/src/main/resources/value.properties 생성

```
<value.properties>
```

```

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

```

9)/src/config/beans.xml 에서 [Namespaces] tab

-목록에서 'context-<http://www.springframework.org/schema/context>' check

-<context:property-placeholder />를 사용하기 위해서

```

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

  <bean id="hello" class="info.javaexpert.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>

```

10)Test

```
-/src/main/java/info.javaexpert.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
```

11)/src/main/resources/value.properties 수정

```
<value.properties>
myname=Spring
myprinter=printer
value1=JUnit
value2=AOP
value3=DI
printer1=stringPrinter
printer2=consolePrinter
```

12)Hello.java 코드 수정

```
-/src/info.javaexpert/Hello.java

package info.javaexpert;

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

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

13)StringPrinter.java 수정

```
package info.javaexpert;

import org.springframework.stereotype.Component;

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

```

461         @Override
462         public void print(String message) {
463             this.buffer.append(message);
464         }
465
466         @Override
467         public String toString(){
468             return this.buffer.toString();
469         }
470     }
471

```

14)beans.xml 수정하기

```

474     <?xml version="1.0" encoding="UTF-8"?>
475     <beans xmlns="http://www.springframework.org/schema/beans"
476           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
477           xmlns:context="http://www.springframework.org/schema/context"
478           xsi:schemaLocation="http://www.springframework.org/schema/beans
479                               http://www.springframework.org/schema/beans/spring-beans.xsd
480                               http://www.springframework.org/schema/context
481                               http://www.springframework.org/schema/context/spring-context-3.2.xsd">
482
483         <context:property-placeholder location="classpath:value.properties "/>
484         <context:component-scan base-package="info.javaexpert" />
485     </beans>

```

15)MainClass.java 수정하기

```

486     package info.javaexpert;
487
488     import java.util.List;
489
490     import org.springframework.context.ApplicationContext;
491     import org.springframework.context.support.GenericXmlApplicationContext;
492
493     public class MainClass {
494         public static void main(String [] args){
495             ApplicationContext ctx = new GenericXmlApplicationContext("classpath:beans.xml");
496
497             Hello hello = (Hello)ctx.getBean("hello");
498             System.out.println(hello.sayHello());
499             hello.print();
500
501             Printer printer = ctx.getBean("stringPrinter", StringPrinter.class);
502             System.out.println(printer.toString());
503
504             List<String> list = hello.getNames();
505             for(String value : list){
506                 System.out.println(value);
507             }
508         }
509     }
510 }
511

```

5. Lab

ApplicationContext.xml --> XML 파일을 이용하는 방법
 -Spring 설정 XML 파일에 property 파일에 대해 명시한다.
 -admin.properties
 -sub_admin.properties

ApplicationConfig --> Java 파일을 이용하는 방법
 -Spring 설정 Java 파일에 property 파일을 명시한다.
 -admin.properties
 -sub_admin.properties

1)/src/main/java/info.javaexper package 생성
 -/src/main/java > right-click > New > Package

-Package name : info.javaexpert

2)/src/main/java/info.javaexpert.AdminConnection.java 생성

```
<AdminConnection.java>
package info.javaexpert;

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() {
        return subAdminId;
    }

    public void setSubAdminId(String subAdminId) {
        this.subAdminId = subAdminId;
    }

    public String getSubAdminPwd() {
        return subAdminPwd;
    }

    public void setSubAdminPwd(String subAdminPwd) {
        this.subAdminPwd = subAdminPwd;
    }
}
```

3)/src/main/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
```

4)/src/main/resources/beans.xml 생성

```
<beans.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">
```

```

591 <context:property-placeholder location="classpath:admin.properties,
592 classpath:sub.admin.properties" />
593
594 <bean id="adminConnection" class="info.javaexpert.AdminConnection">
595     <property name="adminId">
596         <value>${admin.id}</value>
597     </property>
598     <property name="adminPwd">
599         <value>${admin.pwd}</value>
600     </property>
601     <property name="subAdminId">
602         <value>${sub.admin.id}</value>
603     </property>
604     <property name="subAdminPwd">
605         <value>${sub.admin.pwd}</value>
606     </property>
607 </bean>
608 </beans>
609

```

5)/src/main/java/MainClass.java 생성

```

611 <MainClass.java>
612     package info.javaexpert;
613
614     import org.springframework.context.support.AbstractApplicationContext;
615     import org.springframework.context.support.GenericXmlApplicationContext;
616
617     public class MainClass {
618         public static void main(String [] args){
619             AbstractApplicationContext ctx =
620                 new GenericXmlApplicationContext("classpath:beans.xml");
621             AdminConnection connection = ctx.getBean("adminConnection", AdminConnection.class);
622             System.out.println("admin ID : " + connection.getAdminId());
623             System.out.println("admin PWD : " + connection.getAdminPwd());
624             System.out.println("sub admin ID : " + connection.getSubAdminId());
625             System.out.println("sub admin PWD : " + connection.getSubAdminPwd());
626
627             ctx.close();
628         }
629     }
630

```

6. Lab

1)/src/main/resources 두 개의 properties 파일 생성

```

635 <admin.properties>
636     admin.id=javaexpert
637     admin.pwd=12345678
638
639 <sub.admin.properties>
640     sub.admin.id=javasoft
641     sub.admin.pwd=987654321
642

```

2)/src/main/java/info.javaexpert.ApplicationConfig.java>

```

645 <ApplicationConfig.java>
646     package info.javaexpert;
647
648     import org.springframework.beans.factory.annotation.Value;
649     import org.springframework.context.annotation.Bean;
650     import org.springframework.context.annotation.Configuration;
651     import org.springframework.context.support.PropertySourcesPlaceholderConfigurer;
652     import org.springframework.core.io.ClassPathResource;
653     import org.springframework.core.io.Resource;
654
655     @Configuration
656     public class ApplicationConfig {

```

```

657     @Value("${admin.id}")
658     private String adminId;
659     @Value("${admin.pwd}")
660     private String adminPwd;
661     @Value("${sub.admin.id}")
662     private String subAdminId;
663     @Value("${sub.admin.pwd}")
664     private String subAdminPwd;
665
666     @Bean
667     public static PropertySourcesPlaceholderConfigurer Properties(){
668         PropertySourcesPlaceholderConfigurer configurator =
669             new PropertySourcesPlaceholderConfigurer();
670
671         Resource [] locations = new Resource[2];
672         locations[0] = new ClassPathResource("admin.properties");
673         locations[1] = new ClassPathResource("sub.admin.properties");
674         configurator.setLocations(locations);
675
676         return configurator;
677     }
678
679     @Bean
680     public AdminConnection adminConfig(){
681         AdminConnection adminConnection = new AdminConnection();
682         adminConnection.setAdminId(adminId);
683         adminConnection.setAdminPwd(adminPwd);
684         adminConnection.setSubAdminId(subAdminId);
685         adminConnection.setSubAdminPwd(subAdminPwd);
686         return adminConnection;
687     }
688 }

```

3)/src/main/java/info.javaexpert.MainClass1.java

```

692 <MainClass1.java>
693 package info.javaexpert;
694
695 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
696
697 public class MainClass1 {
698     public static void main(String[] args) {
699         AnnotationConfigApplicationContext ctx =
700             new AnnotationConfigApplicationContext(ApplicationConfig.class);
701         AdminConnection conn = ctx.getBean("adminConfig", AdminConnection.class);
702
703         System.out.println("admin ID : " + conn.getAdminId());
704         System.out.println("admin PWD : " + conn.getAdminPwd());
705         System.out.println("sub admin ID : " + conn.getSubAdminId());
706         System.out.println("sub admin PWD : " + conn.getSubAdminPwd());
707     }
708 }

```

7. Profile 속성을 이용한 설정

- 동일한 **Spring Bean**을 여러개 만들어 놓고 상황(환경)에 따라서 적절한 스프링 빈을 사용할 수 있다.
- profile** 속성을 사용한다.
- 역시 **Java** 파일을 이용하는 방법과 **XML** 설정 파일을 이용하는 방법이 있다.

8. Lab

1)Package 생성

- /src/main/java > right-click > New > Package
- Package name : info.javaexpert

2)XML 설정 파일 2개 생성

- /src/main/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="info.javaexpert.ServerInfo">
    <property name="ipNum" value="192.168.56.5" />
    <property name="portNum" value="80" />
  </bean>
</beans>
```

~/src/main/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="info.javaexpert.ServerInfo">
    <property name="ipNum" value="192.168.56.5" />
    <property name="portNum" value="80" />
  </bean>
</beans>
```

3)ServerInfo.java 생성

~/src/main/java/info.javaexpert.ServerInfo.java

```
package info.javaexpert;

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;
  }
}
```

4)MainClass 생성

~/src/main/java/info.javaexpert.MainClass.java

```
package info.javaexpert;
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"
```

```

789         GenericXmlApplicationContext ctx = new GenericXmlApplicationContext();
790         ctx.getEnvironment().setActiveProfiles(config);
791         ctx.load("dev.xml", "run.xml");
792
793         ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
794         System.out.println("IP : " + info.getIpNum());
795         System.out.println("Port : " + info.getPortNum());
796         ctx.close();
797     }
798 }
799

```

5)결과

-입력시 dev를 넣으면 dev환경인 localhost/8080이 나오고, 만일 run이라고 넣으면 192.168.56.5/800이 나온다.

802
803

9. Lab

1)Package 생성

-/src/main/java > right-click > New > Package
-Package name : info.javaexpert

808

2)Java 설정 파일 2개 생성

-/src/main/java > right-click > New > Class
-Class name : ApplicationConfigDev

812

```

813     package info.javaexpert;
814
815     import org.springframework.context.annotation.Bean;
816     import org.springframework.context.annotation.Configuration;
817     import org.springframework.context.annotation.Profile;
818
819     @Configuration
820     @Profile("dev")
821     public class ApplicationConfigDev {
822
823         @Bean
824         public ServerInfo serverInfo(){
825             ServerInfo info = new ServerInfo();
826             info.setIpNum("localhost");
827             info.setPortNum("8080");
828             return info;
829         }
830     }
831

```

-/src/main/java > right-click > New > Class
-Class name : ApplicationConfigRun

834

```

835     package info.javaexpert;
836
837     import org.springframework.context.annotation.Bean;
838     import org.springframework.context.annotation.Configuration;
839     import org.springframework.context.annotation.Profile;
840
841     @Configuration
842     @Profile("run")
843     public class ApplicationConfigRun {
844
845         @Bean
846         public ServerInfo serverInfo(){
847             ServerInfo info = new ServerInfo();
848             info.setIpNum("192.168.56.5");
849             info.setPortNum("80");
850             return info;
851         }
852     }
853

```

854

3)ServerInfo.java 생성

855 -/src/main/java/info.javaexpert.ServerInfo.java

```
856
857 package info.javaexpert;
858
859 public class ServerInfo {
860     private String ipNum;
861     private String portNum;
862     public String getIpNum() {
863         return ipNum;
864     }
865     public void setIpNum(String ipNum) {
866         this.ipNum = ipNum;
867     }
868     public String getPortNum() {
869         return portNum;
870     }
871     public void setPortNum(String portNum) {
872         this.portNum = portNum;
873     }
874 }
```

875 4)MainClass 생성

876 -/src/main/java/info.javaexpert.MainClass.java

```
877
878 package info.javaexpert;
879 import java.util.Scanner;
880
881 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
882
883 public class MainClass {
884     public static void main(String[] args) {
885         Scanner scan = new Scanner(System.in);
886         System.out.print("Select dev or run : ");
887         String config = scan.next(); //"dev" or "run"
888
889         AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext();
890         ctx.getEnvironment().setActiveProfiles(config);
891         ctx.register(ApplicationConfigDev.class, ApplicationConfigRun.class);
892         ctx.refresh();
893
894         ServerInfo info = ctx.getBean("serverInfo", ServerInfo.class);
895         System.out.println("IP : " + info.getIpNum());
896         System.out.println("Port : " + info.getPortNum());
897         ctx.close();
898     }
899 }
900
901 }
```

902 5)결과

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