

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

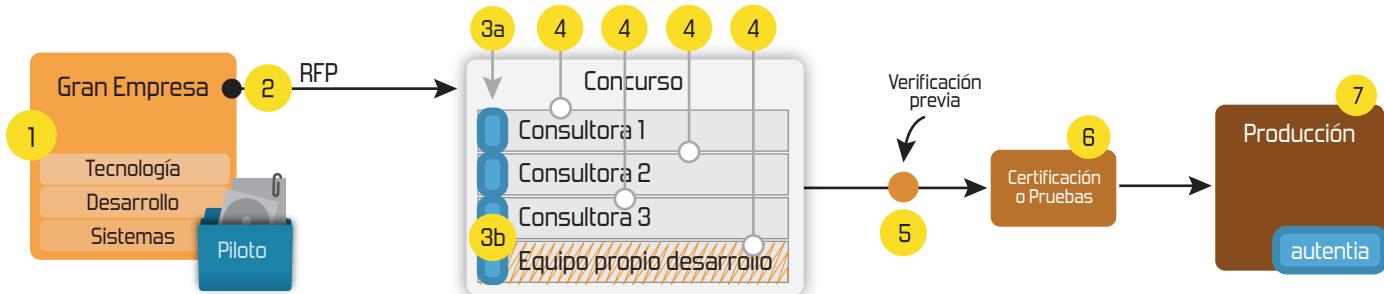
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery

Control de autenticación y acceso (Spring Security)
UDDI

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Web Services
Rest Services
Social SSO
SSO (Cas)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

AdictosAlTrabajo**Terrakas 1x05**
¡¡Ya está en la web!! ;-)
terrakas.comHosting patrocinado por
ENREDADOS

Entra en Adictos a través de [f](#) [t](#)

E-mail
Contraseña

Entrar Deseo registrarme Olvidé mi contraseña

[Inicio](#)[Quiénes somos](#)[Formación](#)[Comparador de salarios](#)[Nuestros libros](#)[Más](#)

» Estás en: Inicio » Tutoriales » Spring Security: haciendo uso de un servidor LDAP embebido.



Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE

[Ver todos los tutoriales del autor](#)**Fecha de publicación del tutorial: 2013-03-04**Tutorial visitado 1 veces [Descargar en PDF](#)

Spring Security: haciendo uso de un servidor LDAP embebido.

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Configuración.
- 4. Test.
- 5. Implementación.
- 6. Referencias.
- 7. Conclusiones.

1. Introducción

Ya hemos tenido oportunidad en adictos de examinar las posibilidades de Spring Security a la hora de gestionar la autenticación y autorización en nuestras aplicaciones; más allá de ser un simple filtro. De hecho hacemos uso intensivo del mismo y forma parte de nuestra arquitectura de referencia.

En este tutorial vamos a ver cómo puede darnos soporte para configurar un servidor LDAP embebido con el objetivo de incluirlo o bien dentro de nuestros tests de integración o para dar soporte a nuestras aplicaciones web sin necesidad de tener un servidor ldap real. Así lo podríamos "levantar" para dar soporte a la autenticación SSO de CAS sin necesidad de disponer de un servidor físico.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 15' (2.4 GHz Intel Core i7, 8GB DDR3 SDRAM).
- Sistema Operativo: Mac OS X Lion 10.7.4
- Spring 3.2.1.RELEASE
- Spring Security 3.1.3.RELEASE

3. Configuración.

Como de costumbre, haciendo uso de maven, lo primero es declarar nuestras dependencias en el fichero pom.xml:

```
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-project_4_0.xsd" version="4.0.0">
2   <modelVersion>4.0.0</modelVersion>
3   <groupId>com.autentia.tutoriales.bpm.ldap</groupId>
4   <artifactId>spring-security-ldap-embedded</artifactId>
5   <packaging>jar</packaging>
6   <version>1.0.0-SNAPSHOT</version>
7   <name>spring-security-ldap-embedded</name>
8
9   <properties>
10    <java.version>1.6</java.version>
11    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
12    <spring.version>3.2.1.RELEASE</spring.version>
13    <spring.security.version>3.1.3.RELEASE</spring.security.version>
14    <slf4j.version>1.6.4</slf4j.version>
15    <log4j.version>1.2.16</log4j.version>
```

Catálogo de servicios Autentia



Síguenos a través de:



Últimas Noticias

» Vendedor: Soy inseguro, filtra o elige por mí: si quieres que te compre.

» Comentando el libro: El arte de pensar, de Rolf Dobelli

» Ya está a la venta mi segundo libro: Planifica tu éxito, de aprendiz a empresario

» Ya esta disponible en eBook mi primer libro: Informática Profesional

» Comentando el libro: La inteligencia reformada, las inteligencias múltiples en el siglo XXI de Howard Gardner

[Histórico de noticias](#)

Últimos Tutoriales

- » Introducción a Guvnor
- » Gestión de expedientes en el ámbito de las Administraciones Públicas (IV): buscando en las forjas una solución.

```

17 </properties>
18
19 <repositories>
20   <repository>
21     <id>com.springsource.repository.bundles.release</id>
22     <name>EBR Spring Release Repository</name>
23     <url>http://repository.springsource.com/maven/bundles/release</url>
24   </repository>
25 </repositories>
26
27 <build>
28   <plugins>
29     <plugin>
30       <groupId>org.apache.maven.plugins</groupId>
31       <artifactId>maven-compiler-plugin</artifactId>
32       <version>2.5.1</version>
33       <configuration>
34         <source>${java.version}</source>
35         <target>${java.version}</target>
36       </configuration>
37     </plugin>
38   </plugins>
39 </build>
40
41 <dependencies>
42   <!-- spring -->
43
44   <dependency>
45     <groupId>org.springframework</groupId>
46     <artifactId>spring-context</artifactId>
47     <version>${spring.version}</version>
48   </dependency>
49
50   <dependency>
51     <groupId>org.springframework</groupId>
52     <artifactId>spring-tx</artifactId>
53     <version>${spring.version}</version>
54   </dependency>
55
56   <!-- spring security -->
57   <dependency>
58     <groupId>org.springframework.security</groupId>
59     <artifactId>spring-security-core</artifactId>
60     <version>${spring.security.version}</version>
61     <exclusions>
62       <exclusion>
63         <groupId>org.springframework</groupId>
64         <artifactId>spring-context</artifactId>
65       </exclusion>
66       <exclusion>
67         <groupId>org.aspectj</groupId>
68         <artifactId>aspectjweaver</artifactId>
69       </exclusion>
70       <exclusion>
71         <groupId>org.springframework</groupId>
72         <artifactId>spring-tx</artifactId>
73       </exclusion>
74       <exclusion>
75         <groupId>org.aspectj</groupId>
76         <artifactId>aspectjrt</artifactId>
77       </exclusion>
78       <exclusion>
79         <groupId>org.springframework</groupId>
80         <artifactId>spring-core</artifactId>
81       </exclusion>
82       <exclusion>
83         <groupId>org.springframework</groupId>
84         <artifactId>spring-aop</artifactId>
85       </exclusion>
86       <exclusion>
87         <groupId>org.springframework</groupId>
88         <artifactId>spring-expression</artifactId>
89       </exclusion>
90       <exclusion>
91         <groupId>org.springframework</groupId>
92         <artifactId>spring-beans</artifactId>
93       </exclusion>
94     </exclusions>
95   </dependency>
96
97   <dependency>
98     <groupId>org.springframework.security</groupId>
99     <artifactId>spring-security-config</artifactId>
100    <version>${spring.security.version}</version>
101    <exclusions>
102      <exclusion>
103        <groupId>org.springframework</groupId>
104        <artifactId>spring-context</artifactId>
105      </exclusion>
106      <exclusion>
107        <groupId>org.aspectj</groupId>
108        <artifactId>aspectjweaver</artifactId>
109      </exclusion>
110      <exclusion>
111        <groupId>org.springframework</groupId>
112        <artifactId>spring-tx</artifactId>
113      </exclusion>
114      <exclusion>
115        <groupId>org.aspectj</groupId>
116        <artifactId>aspectjrt</artifactId>
117      </exclusion>
118      <exclusion>
119        <groupId>org.springframework</groupId>

```

- » Gestión de expedientes en el ámbito de las Administraciones Públicas (III): BPM y la gestión de procesos de negocio.
- » jBPM5: usando nuestra propia base de datos.
- » AngularJS: primeros pasos.

Últimos Tutoriales del Autor

- » Introducción a Guvnor
- » Gestión de expedientes en el ámbito de las Administraciones Públicas (IV): buscando en las forjas una solución.
- » Gestión de expedientes en el ámbito de las Administraciones Públicas (III): BPM y la gestión de procesos de negocio.
- » jBPM5: usando nuestra propia base de datos.
- » Gestión de expedientes en el ámbito de las Administraciones Públicas (II): requisitos.

Últimas ofertas de empleo

- 2011-09-08 Comercial - Ventas - MADRID.
- 2011-09-03 Comercial - Ventas - VALENCIA.
- 2011-08-19 Comercial - Compras - ALICANTE.
- 2011-07-12 Otras Sin catalogar - MADRID.
- 2011-07-06 Otras Sin catalogar - LUGO.



Jose Manuel Sánchez
sanchezsanchez

adictosaltrabajo Introducción a Guvnor – kcy.me/g8hs como repositorio central de recursos dentro de nuestro ecosistema bpm #jbpm
2 days ago · reply · retweet · favorite

adictosaltrabajo Gestión de expedientes en el ámbito de las Administraciones Públicas (IV): buscando en las forjas una solución. – kcy.me/g0de
6 days ago · reply · retweet · favorite

adictosaltrabajo Gestión de expedientes en el ámbito de las Administraciones Públicas (III): BPM y la gestión de procesos de negocio – kcy.me/fyzp
6 days ago · reply · retweet · favorite

[Join the conversation](#)

```

120             <artifactId>spring-core</artifactId>
121         </exclusion>
122     <exclusion>
123         <groupId>org.springframework</groupId>
124         <artifactId>spring-aop</artifactId>
125     </exclusion>
126     <exclusion>
127         <groupId>org.springframework</groupId>
128         <artifactId>spring-expression</artifactId>
129     </exclusion>
130     <exclusion>
131         <groupId>org.springframework</groupId>
132         <artifactId>spring-beans</artifactId>
133     </exclusion>
134   </exclusions>
135 </dependency>
136
137 <dependency>
138     <groupId>org.springframework.security</groupId>
139     <artifactId>spring-security-ldap</artifactId>
140     <version>${spring.security.version}</version>
141     <exclusions>
142         <exclusion>
143             <groupId>org.springframework</groupId>
144             <artifactId>spring-context</artifactId>
145         </exclusion>
146         <exclusion>
147             <groupId>org.springframework</groupId>
148             <artifactId>spring-tx</artifactId>
149         </exclusion>
150         <exclusion>
151             <groupId>org.springframework</groupId>
152             <artifactId>spring-core</artifactId>
153         </exclusion>
154         <exclusion>
155             <groupId>org.springframework</groupId>
156             <artifactId>spring-beans</artifactId>
157         </exclusion>
158     </exclusions>
159 </dependency>
160 <dependency>
161     <groupId>org.apache.directory.server</groupId>
162     <artifactId>apacheds-all</artifactId>
163     <version>1.5.5</version>
164 </dependency>
165
166 <!-- logging -->
167 <dependency>
168     <groupId>org.slf4j</groupId>
169     <artifactId>slf4j-api</artifactId>
170     <version>${slf4j.version}</version>
171 </dependency>
172 <dependency>
173     <groupId>org.slf4j</groupId>
174     <artifactId>slf4j-log4j12</artifactId>
175     <version>${slf4j.version}</version>
176 </dependency>
177 <dependency>
178     <groupId>log4j</groupId>
179     <artifactId>log4j</artifactId>
180     <version>${log4j.version}</version>
181 </dependency>
182
183 <!-- Test -->
184 <dependency>
185     <groupId>junit</groupId>
186     <artifactId>junit</artifactId>
187     <version>4.8.2</version>
188     <scope>test</scope>
189 </dependency>
190
191 <dependency>
192     <groupId>org.springframework</groupId>
193     <artifactId>spring-test</artifactId>
194     <version>${spring.version}</version>
195     <scope>test</scope>
196 </dependency>
197
198 </dependencies>
199
200 </project>

```

El servidor que usaremos será Apache Directory que se alimentará en su arranque de un fichero ldif (autentia-identity-repository.ldif), con la siguiente información sobre nuestra organización:

```

1 version: 1
2
3 dn: o=autentia
4 objectClass: organization
5 objectClass: extensibleObject
6 objectClass: top
7 o: autentia
8
9 dn: ou=users,o=autentia
10 objectClass: extensibleObject
11 objectClass: organizationalUnit
12 objectClass: top
13 ou: users
14
15 dn: ou=groups,o=autentia
16 objectClass: extensibleObject
17 objectClass: organizationalUnit
18 objectClass: top

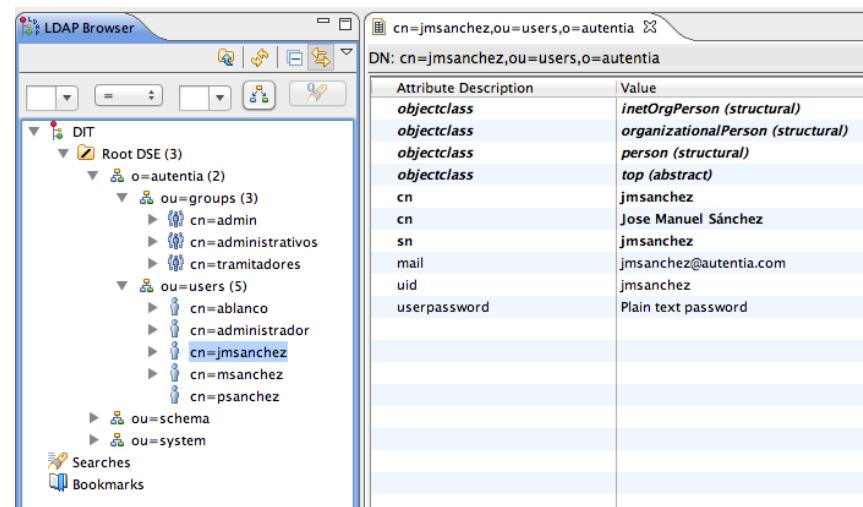
```

```

19 ou: groups
20
21 dn: cn=administrativos,ou=groups,o=autentia
22 objectClass: groupOfUniqueNames
23 objectClass: top
24 objectClass: group
25 cn: administrativos
26 uniqueMember: cn=jmsanchez,ou=users,o=autentia
27 uniqueMember: cn=psanchez,ou=users,o=autentia
28
29 dn: cn=tramitadores,ou=groups,o=autentia
30 objectClass: groupOfUniqueNames
31 objectClass: top
32 objectClass: group
33 cn: tramitadores
34 uniqueMember: cn=ablanco,ou=users,o=autentia
35 uniqueMember: cn=msanchez,ou=users,o=autentia
36
37 dn: cn=admin,ou=groups,o=autentia
38 objectClass: groupOfUniqueNames
39 objectClass: top
40 objectClass: group
41 cn: admin
42 uniqueMember: cn=administrador,ou=users,o=autentia
43
44 dn: cn=jmsanchez,ou=users,o=autentia
45 objectClass: organizationalPerson
46 objectClass: person
47 objectClass: inetOrgPerson
48 objectClass: top
49 cn: Jose Manuel Sánchez
50 sn: jmsanchez
51 uid: jmsanchez
52 mail: jmsanchez@autentia.com
53 userPassword:: cGFzcw==
54
55 dn: cn=psanchez,ou=users,o=autentia
56 objectClass: organizationalPerson
57 objectClass: person
58 objectClass: inetOrgPerson
59 objectClass: top
60 cn: Pablo Sánchez
61 sn: psanchez
62 uid: psanchez
63 mail: psanchez@autentia.com
64 userPassword:: cGFzcw==
65
66 dn: cn=msanchez,ou=users,o=autentia
67 objectClass: organizationalPerson
68 objectClass: person
69 objectClass: inetOrgPerson
70 objectClass: top
71 cn: Mario Sánchez
72 sn: msanchez
73 uid: msanchez
74 mail: msanchez@autentia.com
75 userPassword:: cGFzcw==
76
77 dn: cn=ablanco,ou=users,o=autentia
78 objectClass: organizationalPerson
79 objectClass: person
80 objectClass: inetOrgPerson
81 objectClass: top
82 cn: Alfonso Blanco
83 sn: ablanco
84 uid: ablanco
85 mail: ablanco@autentia.com
86 userPassword:: cGFzcw==
87
88 dn: cn=administrador,ou=users,o=autentia
89 objectClass: organizationalPerson
90 objectClass: person
91 objectClass: inetOrgPerson
92 objectClass: top
93 cn: admin
94 sn: admin
95 uid: administrador
96 userPassword:: cGFzcw==

```

Tenemos una pequeña jerarquía, con dos grupos de usuarios: administrativos y tramitadores (además del grupo de administradores) y varios usuarios asociados a los mismos que montará una estructura como la siguiente:



Nota: para obtener esta captura he puesto un punto de ruptura en el test que veremos en el siguiente punto y he comprobado el acceso al directorio, por el puerto configurado, con la herramienta Apache Directory Studio

La configuración del servidor LDAP la incluimos en un fichero de configuración Spring, con un contenido similar al siguiente (src/main/resources/spring-config/ldapServer.xml):

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:security="http://www.springframework.org/schema/security"
5   xsi:schemaLocation="http://www.springframework.org/schema/beans
6     http://www.springframework.org/schema/beans/spring-beans.xsd
7     http://www.springframework.org/schema/security
8     http://www.springframework.org/schema/security/spring-security.xsd">
9
10   <security:ldap-server ldif="classpath:autentia-identity-repository.ldif" root="o=aut"
11
12 </beans>
```

En el propio espacio de nombres del contexto de seguridad disponemos de una configuración para ldap donde configuramos el fuente ldif con el contenido de nuestra organización, el nodo raíz para acceder al mismo y el puerto, a través del cuál podemos acceder al mismo.

4. Test.

Vamos a escribir, con el soporte de Spring, un test que compruebe el acceso y recuperación de usuarios y grupos del directorio:

```

1 package com.autentia.tutoriales.spring.security.ldap;
2
3 import static org.junit.Assert.assertEquals;
4
5 import java.util.List;
6
7 import org.junit.Test;
8 import org.junit.runner.RunWith;
9 import org.springframework.beans.factory.annotation.Autowired;
10 import org.springframework.test.context.ContextConfiguration;
11 import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
12
13
14 @RunWith(SpringJUnit4ClassRunner.class)
15 @ContextConfiguration({"classpath:spring-config/ldapServer.xml", "classpath:spring-config"})
16 public class PersonRepositoryIntegrationTest {
17
18   @Autowired
19   private PersonRepository personRepository;
20
21   @Test
22   public void shouldFindUserById() {
23     final String userid = "jmsanchez";
24     final List<Person> persons = personRepository.getByUID(userid);
25     assertEquals(1, persons.size());
26     assertEquals("Jose Manuel Sánchez", persons.get(0).getName());
27   }
28
29   @Test
30   public void shouldFindUsersByGroupId() {
31     final String groupId = "administrativos";
32     final List<Person> persons = personRepository.getByGroupUID(groupId);
33     assertEquals(2, persons.size());
34   }
35 }
```

En este punto nuestro test estará en ROJO (errores de compilación), porque:

- no disponemos de la clase PersonRepository, que nos proporcionará acceso al repositorio de usuarios o personas,
- no tenemos la clase Person, que tendrá las propiedades de un usuario, y
- hemos separado la configuración del servidor de la configuración del contexto para acceder al mismo (aún no hemos

visto el contenido del fichero de configuración `ldapContext.xml`).

Veamos la interfaz de servicio que hemos pensado para el repositorio de personas

```

1 package com.autentia.tutoriales.spring.security.ldap;
2
3 import java.util.List;
4
5 public interface PersonRepository {
6
7     List<Person> getByUID(String userId);
8
9     List<Person> getByGroupUID(String groupId);
10
11 }
```

Y ahora el contenido del POJO que tendrá las propiedades de un usuario:

```

1 package com.autentia.tutoriales.spring.security.ldap;
2
3 public class Person {
4
5     private String uid;
6
7     private String name;
8
9     private String email;
10
11     public String getUid() {
12         return uid;
13     }
14
15     public void setUid(String uid) {
16         this.uid = uid;
17     }
18
19     public String getName() {
20         return name;
21     }
22
23     public void setName(String name) {
24         this.name = name;
25     }
26
27     public String getEmail() {
28         return email;
29     }
30
31     public void setEmail(String email) {
32         this.email = email;
33     }
34 }
```

Ahora no tendremos errores de compilación, pero fallará la ejecución del test:

5. Implementación.

Veamos el contenido que proponemos para el repositorio de usuarios:

```

1 package com.autentia.tutoriales.spring.security.ldap;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 import javax.naming.NamingEnumeration;
7 import javax.naming.NamingException;
8 import javax.naming.directory.Attributes;
9
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.ldap.core.AttributesMapper;
12 import org.springframework.ldap.core.LdapTemplate;
13
14 public class PersonLdapRepository implements PersonRepository{
15
16     private LdapTemplate ldapTemplate;
17
18     @Autowired
19     public PersonLdapRepository(LdapTemplate ldapTemplate){
20         this.ldapTemplate = ldapTemplate;
21     }
22
23     @SuppressWarnings("unchecked")
24     @Override
25     public List<Person> getByUID(String userId) {
26         return ldapTemplate.search("", "(uid=" + userId + ")",
27             personMapper());
27     }
28
29     @Override
30     public List<Person> getByGroupUID(String groupId) {
31         final List<Person> persons = new ArrayList<Person>();
32         @SuppressWarnings("unchecked")
33         final List<String> userIds = (List<String>) ldapTemplate.search(
34             "", "(&(objectCl
35             for (String userId : user_ids) {
36                 persons.add(getByUID(userId).get(0));
37             }
38         return persons;
39     }
40
41     private AttributesMapper personMapper() {
```

```

41     return new AttributesMapper() {
42
43         @Override
44         public Person mapFromAttributes(final Attributes attributes) throws NamingException {
45             final Person person = new Person();
46             person.setUid((String) attributes.get("uid").get());
47             person.setName((String) attributes.get("cn").get());
48             person.setEmail((String) attributes.get("mail").get());
49             return person;
50         }
51     };
52 }
53
54 private AttributesMapper groupMapper() {
55     return new AttributesMapper() {
56
57         @Override
58         public List<String> mapFromAttributes(final Attributes attributes) throws NamingException {
59             final List<String> userids = new ArrayList<String>();
60             final NamingEnumeration<?> uniquemembers = attributes.get("uniqueMember");
61             while (uniquemembers.hasMore()) {
62                 String userId = uniquemembers.next().toString().split(",")[0].split(
63                     userIds.add(userId);
64                 }
65             }
66         }
67     };
68 }
69 }
70 }
```

Básicamente, se apoya en una clase de plantilla de Spring para realizar las consultas al servidor ldap y se encarga de mapear el resultado de las consultas en un objeto de tipo Person (lo que haría un típico Dao).

La configuración que nos resta es la del fichero (src/main/resources/spring-config/ldapContext.xml) que va a configurar el contexto de acceso al servidor ldap que tenemos levantado para el entorno de tests, lo va a asociar a la plantilla de spring y ésta será inyectada en nuestro servicio.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:security="http://www.springframework.org/schema/security"
5   xsi:schemaLocation="http://www.springframework.org/schema/beans
6       http://www.springframework.org/schema/spring-beans.xsd
7       http://www.springframework.org/schema/security
8       http://www.springframework.org/schema/security/spring-security.xsd">
9
10 <bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource"
11   init-method="afterPropertiesSet">
12   <property name="url" value="ldap://localhost:9898" />
13   <property name="base" value="o=autentia" />
14   <property name="userDn" value="uid=admin,ou=system" />
15   <property name="password" value="secret" />
16 </bean>
17
18 <bean id="ldapTemplate" class="org.springframework.ldap.core.LdapTemplate"
19   init-method="afterPropertiesSet">
20   <constructor-arg ref="contextSource"/>
21 </bean>
22
23 <bean id="personRepository" class="com.autentia.tutoriales.spring.security.ldap.PersonRepository"
24   <constructor-arg ref="ldapTemplate"/>
25 </bean>
26
27 </beans>
```

6. Referencias.

- <http://static.springsource.org/spring-security/site/docs/3.0.x/reference/ldap.html>
- <http://static.springsource.org/spring-ldap/docs/1.3.x/reference/pdf/spring-ldap-reference.pdf>
- <http://krams915.blogspot.com.es/2011/01/spring-security-mvc-using-embedded-ldap.html>

7. Conclusiones.

Como siempre, con el soporte de Spring, lo difícil se termina convirtiendo en algo fácil y vamos allanando el terreno para configurar un sistema de identidad único en nuestro entorno.

Un saludo.

Jose

jmsanchez@autentia.com

A continuación puedes evaluarlo:

Regístrate para evaluarlo



Por favor, vota +1 o compártelo si te pareció interesante



Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

» **Regístrate** y accede a esta y otras ventajas «



Esta obra está licenciada bajo licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5

Copyright 2003-2013 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) | [Contacto](#)

