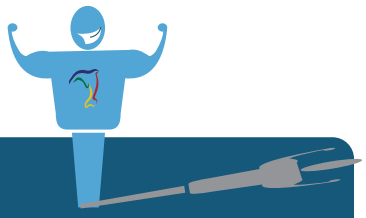
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

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

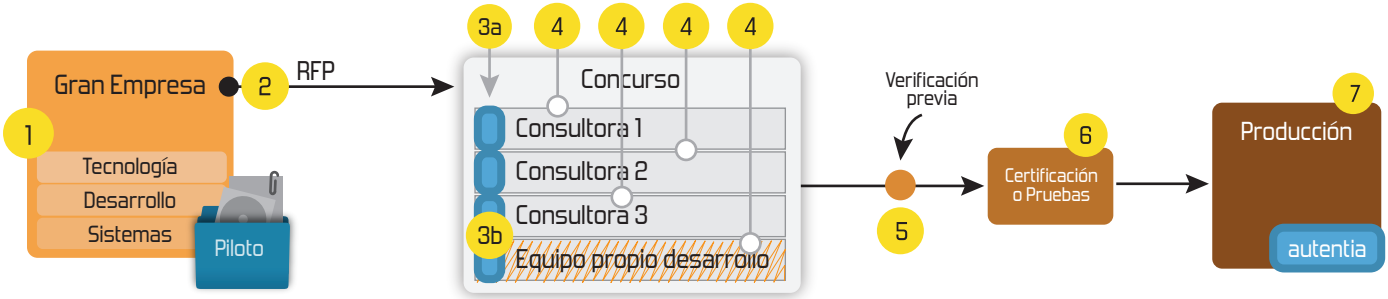
1. Desarrollo de componentes y proyectos a medida



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

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

Control de autenticación y acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

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

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)

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

E-mail:

Contraseña:

Deseo registrarme Entrar

He olvidado mis datos de acceso

DESARROLLADO POR:
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

Fecha de publicación del tutorial: 2009-02-26

Indexación y recuperación de documentos en Apache Solr haciendo uso del api para Java.

0. Índice de contenidos.

1. Introducción

Después de la introducción a Apache Solr, en este tutorial vamos a estudiar cómo indexar documentos y recuperarlos del índice, haciendo uso del api de java solrj. No vamos a entrar aún en la configuración de los campos y tipos de datos de Solr, ni en la configuración de los analizadores para la indexación y recuperación; vamos a usar los campos y los analizadores por defecto definidos para el servidor de ejemplo.

También vamos a aprovechar el tutorial para mostrar como crear un proyecto y añadir dependencias con el soporte de m2eclipse.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

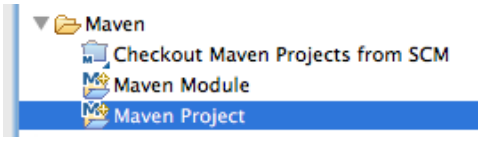
- Hardware: Portátil MacBook Pro 17' (2.93 GHz Intel Core 2 Duo, 4GB DDR3 SDRAM).

- Sistema Operativo: Mac OS X Snow Leopard 10.6.7
- Apache Solr 3.1.
- Eclipse Helios SR2 con m2eclipse
- Junit 4.8.2

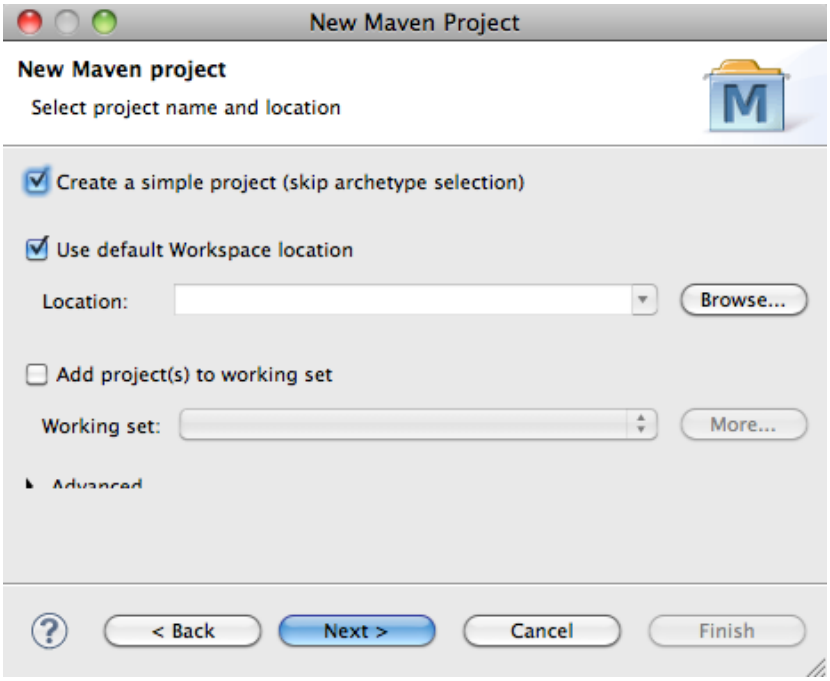
3. Un proyecto con soporte para Solrj.

Haciendo uso del plugin de maven para elcipse, m2eclipse, pulsamos sobre new > project:

Seleccionamos Maven > Maven Project:



Seleccionamos la creación de un proyecto simple:



Asignamos las características que describen nuestro proyecto y que se trasladarán al pom.xml:

Session TimeOut en JSF2 con el soporte de Primefaces.

Cambiando el plugin de eclipse para Maven, de IAM a m2eclipse.

Primeros pasos con github: subir un proyecto al repositorio.

Últimos Tutoriales del Autor

Introducción a Apache Solr.

Session TimeOut en JSF2 con el soporte de Primefaces.

Primeros pasos con github: subir un proyecto al repositorio.

Habilitar autocompletado de etiquetas para JSF en un proyecto Eclipse gestionado por Maven.

Listener del ciclo de vida de JSF, en JSF2 y con el soporte de FacesTrace (Primefaces)

Síguenos a través de:

Últimas ofertas de empleo

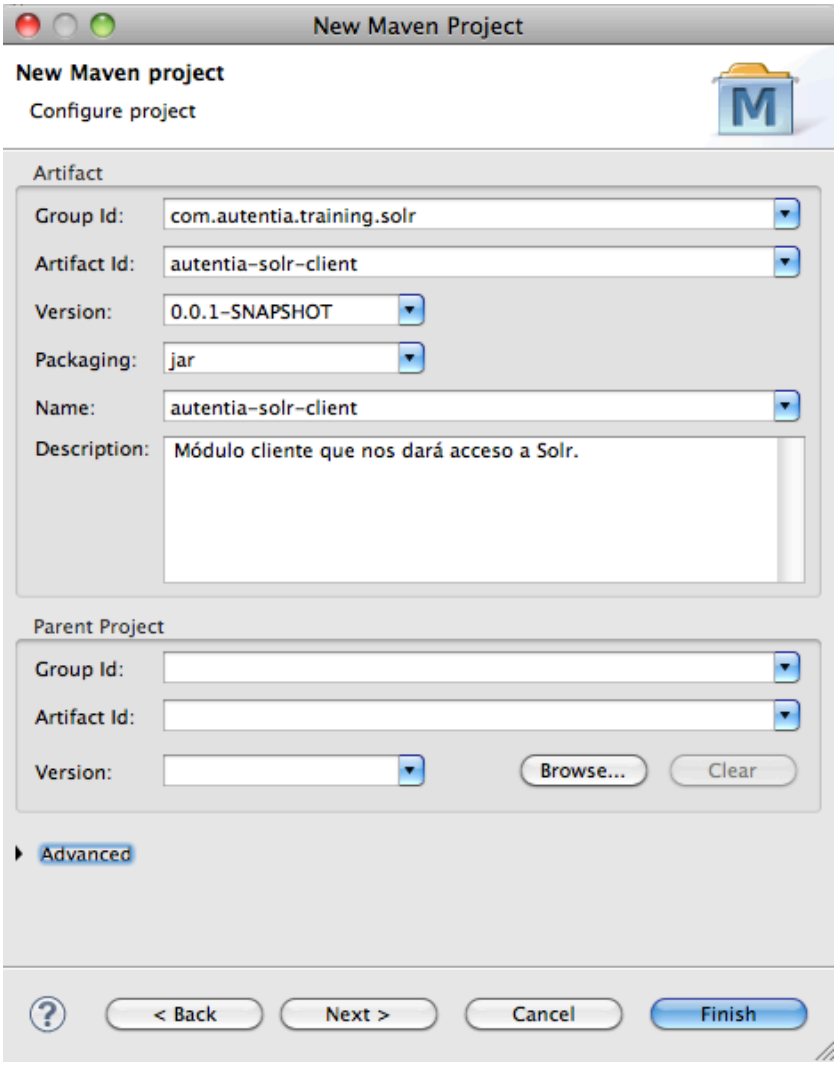
2011-03-02
T. Información - Analista / Programador - MALAGA.

2011-02-24
T. Información - Especialista CRM - MADRID.

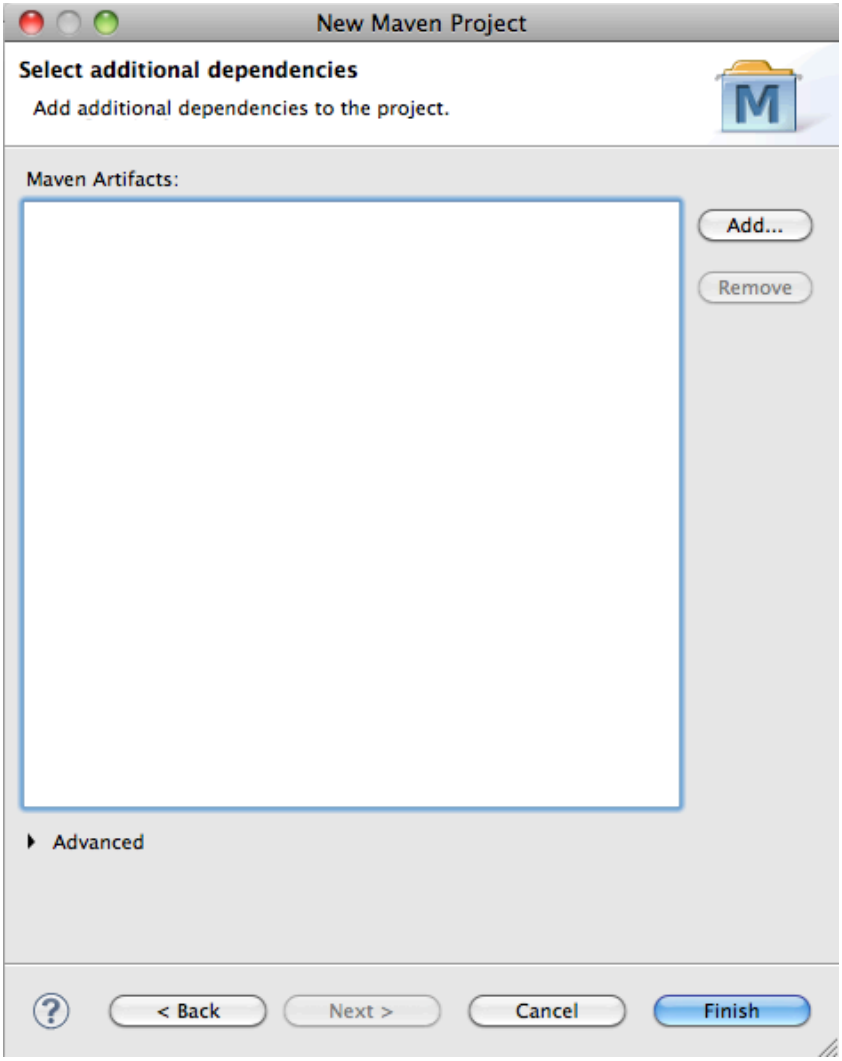
2011-02-16
Marketing - Experto en Marketing - CADIZ.

2011-02-08
Comercial - Ventas - CADIZ.

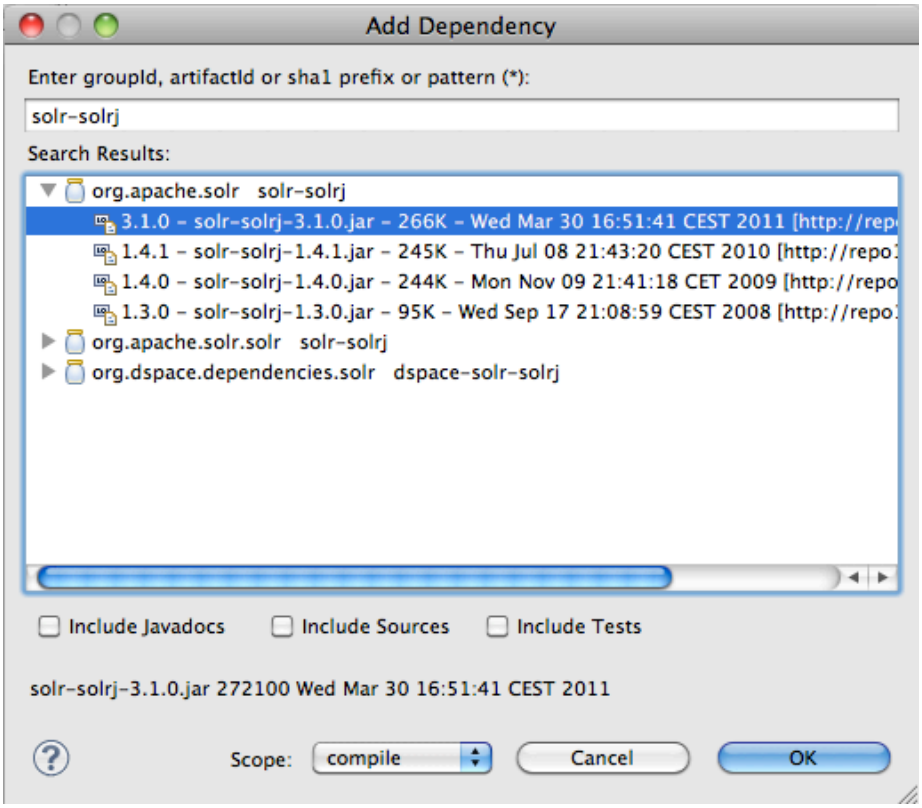
2011-01-28
Comercial - Ventas - SEVILLA.



Y podemos asignar las librerías con las que vamos a trabajar, pulsando sobre add:



Podemos realizar búsquedas, como si estuviésemos trabajando con mvnrepository.com



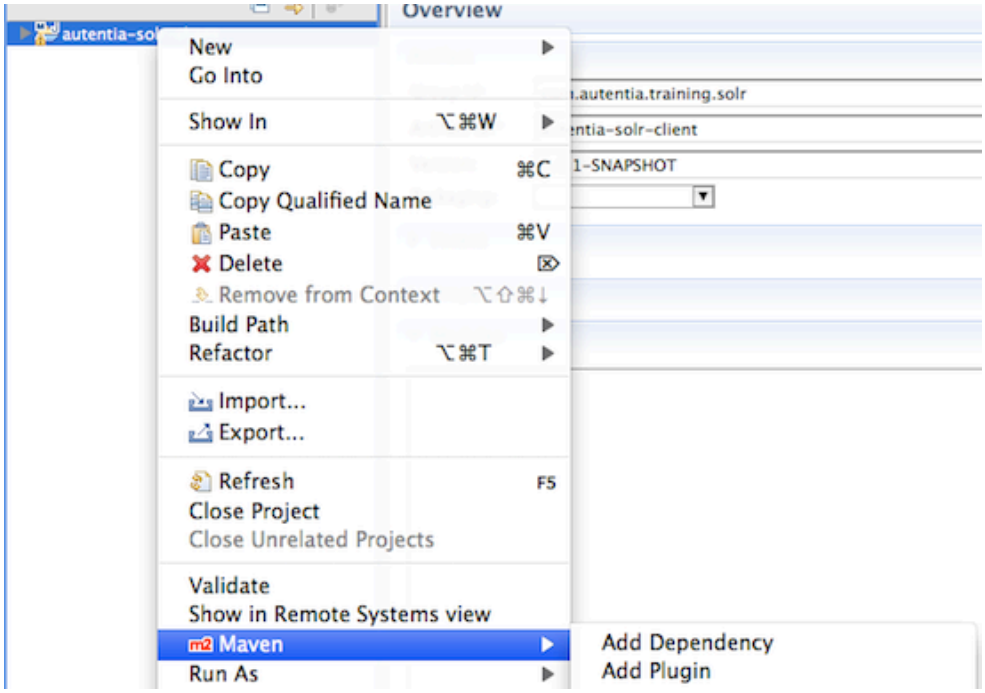
Tras finalizar el wizard, deberíamos tener un proyecto en el workspace y en el pom.xml la dependencia de la librería de solrj.

```xml
<dependency>
  <groupId>org.apache.solr</groupId>
  <artifactId>solr-solrj</artifactId>
  <version>3.1.0</version>
</dependency>
```

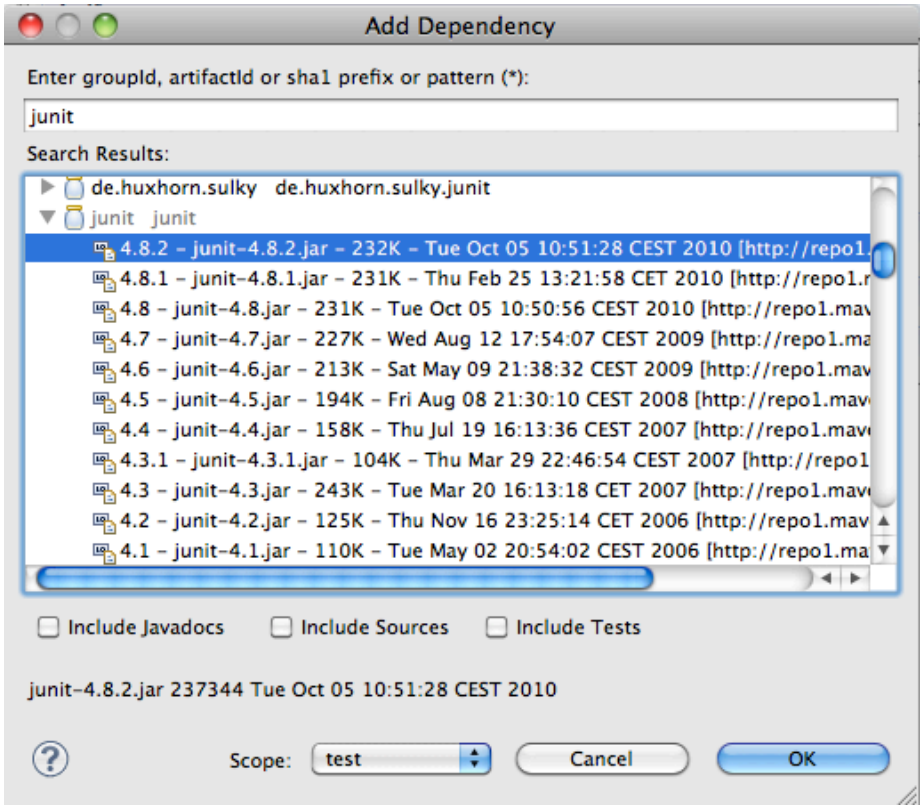
Una vez creado el proyecto también podemos añadir dependencias de librerías con el soporte de m2eclipse, de hecho vamos a añadir la de

pendencia a la librería de junit en el ámbito de test.

Pulsando sobre el proyecto, botón derecho > Maven > Add Dependency:



Buscamos la librería y la añadimos:



Solo nos queda añadir la dependencia a la librería de sl4j, la podemos buscar o añadir a mano en el pom.xml; en cualquier caso que sea la versión 1.5.5, porque a partir de la 1.5.6 modificaron la visibilidad del campo SINGLETON de la clase org.slf4j.impl.StaticLoggerBinder y la forma que la que se inicializan los loggers en la librería solrj no es compatible con dicha modificación.

```xml
<dependency>
    <groupid>org.slf4j</groupid>
    <artifactid>slf4j-simple</artifactid>
```

```
    <version>1.5.5</version>
</dependency>
```

4. Recuperación.

Como de la introducción a Apache Solr ya tenemos cargados productos en Solr, lo primero que vamos a hacer es intentar recuperarlos.

Para ello, escribimos el siguiente test:

```
public class SolrIndexerTest {

    private static CommonsHttpSolrServer server;

    @BeforeClass
    public static void init() throws MalformedURLException{
        server = new CommonsHttpSolrServer("http://localhost:8983/solr");
        server.setParser(new XMLResponseParser());
    }

    @Test
    public void retrieveDocumentsFromSolr() throws SolrServerException{
        final SolrDocumentList results = findByName("ipod");
        Assert.assertEquals(3, results.size());
        Assert.assertEquals("iPod & iPod Mini USB 2.0 Cable",
results.get(0).get("name"));
    }

    private SolrDocumentList findByName(String name) throws
SolrServerException{
        final SolrQuery query = new SolrQuery();
        query.setQuery("name:" + name);
        return server.query(query).getResults();
    }

}
```

Hay 3 productos que responden al término ipod y el primero tiene este nombre "iPod & iPod Mini USB 2.0 Cable".

Sin Solr levantado y escuchando peticiones por el puerto 8993, obtendremos una:

`org.apache.solr.client.solrj.SolrServerException: java.net.ConnectException: Connection refused`

Si necesitásemos añadir paginación a la consulta bastaría con incluir la invocación a los siguientes métodos:

```
query.setStart(0);
query.setRows(100);
```

5. Indexación.

Para indexar un nuevo producto en Solr podemos añadir un test como el que sigue:

```
@Test
public void addDocumentInSolr() throws SolrServerException, IOException{
    final SolrInputDocument product = new SolrInputDocument();
    product.addField("id", "99");
    product.addField("name", "Las reglas no escritas para triunfar en la
empresa. 2ª EDICIÓN ACTUALIZADA.");
    product.addField("author", "Roberto Canales");
    server.add(Arrays.asList(product));
    server.commit();

    Assert.assertEquals(1, findByName("Las reglas no escritas para triunfar
en la empresa").size());

    server.deleteById("99");
    server.commit();

    Assert.assertEquals(0, findByName("Las reglas no escritas para triunfar
en la empresa").size());
}
```

Al igual que vimos con el formato del xml, aquí añadimos campos vinculándolos a un nombre y proporcionando un cotenido textual.

Buscando la atomicidad, borramos el producto antes de salir del método de test.

6. Con el soporte de la anotación @Field.

Solr nos permite hacer uso de la anotación org.apache.solr.client.solrj.beans.Field para declarar atributos de nuestros POJOs como campos indexables. Para ello, basta con añadir un metadato adicional a nuestras clases del modelo de datos.

En una entidad gestionada por Hibernate, quedaría como sigue:

```
...
@Entity
public class Product implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Field
    private Long id;

    @NotNull
    @Column(nullable=false)
    private String ean;

    @Field
    private String name;

    private boolean active;

    @Field
    private BigDecimal price;

    // ... getters & setters

}
```

Para indexar nuestra entidad bastaría con invocar al método addBean de la instancia del server, como se muestra a continuación:

```
final Product product = new Product();
product.setId(9L);
product.setEan("111111111111");
product.setName("Solr 3.1 Enterprise Search Server");

server.addBean(product);
server.commit();
```

Para recuperar directamente un listado de productos, bastaría con invocar al método getBeans, pasándole la clase a convertir:

```
final SolrQuery query = new SolrQuery();
query.setQuery("name:" + name);
final List<Product> products = server.query(query).getBeans(Product.class);
```

Hay que tener en cuenta que con este último método recuperaríamos los productos poblados con los campos que se almacenan en Solr, no tendría toda la información de un producto (sino la almacenamos en Solr) y la entidad no estaría en el contexto de Hibernate.

Por otro lado, nos veríamos en la necesidad de hacer uso de los tipos básicos de campos que maneja Solr, no podemos indexar un tipo complejo (una instancia de una clase o un tipo que no este manejado en Solr, por defecto, como BigDecimal, a no ser que creemos un tipo de dato personalizado).

7. Referencias.

- http://wiki.apache.org/solr/Solrj

8. Conclusiones.

Ya conocemos algo más sobre el api de java de Solr y podemos comenzar a trabajar indexando y recuperando contenido.

Para todo el entorno de pruebas hemos estado usando un Solr real, el de ejemplo de la distribución, pero real. Lo siguiente será configurar y levantar un Solr embebido solo para el entorno de tests.

Un saludo.

Jose

jmsanchez@autentia.com

(Sólo para usuarios registrados)

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

Comentarios