

¿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

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

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

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

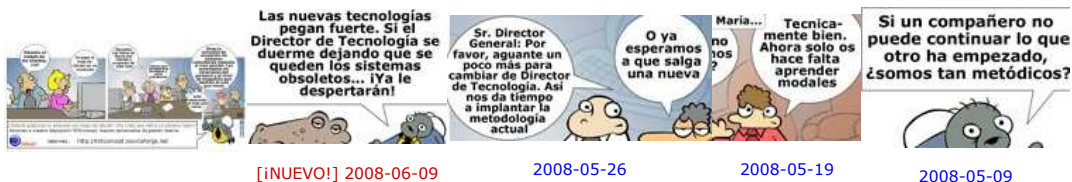


[Inicio](#) [Quienes somos](#) [Tutoriales](#) [Formación](#) [Empleo](#) [Colabora](#) [Comunidad](#) [Libro de Visitas](#) [Comic](#)

NUEVO ¿Quieres saber cuánto ganas en relación al mercado? pincha aquí...

[Ver cursos que ofrece Autentia](#)

[Descargar comics en PDF y alta resolución](#)



[NUEVO!] 2008-06-09

2008-05-26

2008-05-19

2008-05-09

Estamos escribiendo un libro sobre la profesión informática y estas viñetas formarán parte de él. Puedes opinar en la sección [comic](#).

Tutorial desarrollado por



Alejandro Pérez García

Alejandro es socio fundador de Autentia y nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.

Ingeniero en Informática

Si te gusta lo que ves, puedes contratarle para impartir **cursos presenciales** en tu empresa o para **ayudarte en proyectos** (Madrid). Puedes encontrarme en [Autentia](#)

Catálogo de servicios de Autentia

[Descargar \(6,2 MB\)](#)

[Descargar en versión comic \(17 MB\)](#)

[AdictosAlTrabajo.com](#) es el Web de difusión de conocimiento de [Autentia](#).



[Catálogo de cursos](#)

Descargar este documento en formato PDF: [hibernateSearch.pdf](#)

Fecha de creación del tutorial: 2008-06-13

Toda la potencia de un buscador como Google en tu base de datos, gracias a Hibernate Search

Creación: 13-06-2008

Índice de contenidos

- [1. Introducción](#)
- [2. Entorno](#)
- [3. Configuración del entorno de desarrollo](#)
- [4. Configuración de Hibernate Search](#)
- [5. Construyendo un POJO que sea "indexable"](#)
- [6. Como hacer consultas en nuestros índices](#)
- [7. Recursos](#)
- [8. Conclusiones](#)
- [9. Sobre el autor](#)

1. Introducción

Sí, has oído bien, en este tutorial vamos a ver una introducción de como montar un sistema para poder hacer búsquedas textuales tipo Google sobre la información que tenemos almacenada en nuestra base de datos.

Para ello vamos a utilizar **Hibernate Search** (<http://www.hibernate.org/410.html>). Este proyecto hace algún tiempo que vio luz, y lo que hace es combinar la potencia de **Hibernate** (<http://www.hibernate.org>) para la gestión de bases de datos, con la potencia de **Apache Lucene** (<http://lucene.apache.org/java/docs/index.html>) como motor de búsquedas textuales basado en índices inversos.

Gracias a Hiberante Search conseguimos unificar estas dos grandes herramientas unificando manejo e interfaces. Y resulta trivial dotar de esta funcionalidad a nuestras aplicaciones.

Tal como nos cuenta la gente de Hibernate Search en su página principal, algunas de las principales ventajas son:

- Solventa el problema del desajuste de estructuras entre Hibernate y Lucene. Es decir, se encarga de hacer la traducción entre nuestros objetos y los índices de Lucene.
- Acaba con los problemas de duplicidad, manejando los índices, manteniendo sincronizados los cambios con la base de datos, y optimizando el acceso al índice, de forma transparente para el desarrollador.

[Anuncios Google](#) [Hibernate Libraries](#) [Hibernate ORM](#) [Hibernate Project](#) [Hibernate Mapping](#) [Hibernate Eclipse](#)

Catálogo de servicios Autentia (PDF 6,2MB)



[En formato comic...](#)



☐ Web

☒ [www.adictosaltrabajo.com](#)

Últimos tutoriales

2008-06-14
[Hibernate Validator, y como definir las validaciones sobre los objetos de negocio](#)

2008-06-13
[Toda la potencia de un buscador como Google en tu base de datos, gracias a Hibernate Search](#)

2008-06-09
[Arquetipos de maven: cómo crear, distribuir y generar proyectos con JSF e ICEfaces, JBoss y EJB3](#)

2008-06-04
[Desarrollos Web en PHP con AppServ 2.5.6 y Eclipse PDT](#)

2008-06-02
[Ficheros de mapeo de Hibernate desde las clases](#)

2008-06-02
[Un vistazo a Gantt Project](#)

2008-05-29
[Manejar presentaciones con UNO](#)

2008-05-25
[Composición de música con TUXGUITAR](#)

2008-05-14
[Spring + Hibernate + Anotaciones = Desarrollo Rápido en Java](#)

2008-05-06
[J2ME. Internacionalización de aplicaciones para móviles](#)

Últimas ofertas de empleo

- Unifica los APIs, de manera que se pueden hacer búsquedas en el índice y recuperar los objetos pertinentes como si se tratara de una consulta normal de Hiberante (además siempre tenemos la opción de acceder directamente al API Lucene, si fuera necesario).

Pero bueno, empecemos ya que me estoy poniendo nervioso ;)

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Asus G1 (Core 2 Duo a 2.1 GHz, 2048 MB RAM, 120 GB HD).
- Nvidia GEFORCE GO 7700
- Sistema Operativo: GNU / Linux, Debian (unstable), Kernel 2.6.24, KDE 3.5
- Java Sun 1.6.0_06
- Hibernate 3.2.6.ga
- Hibernate Search 3.0.1 GA
- JUnit 4.4

3. Configuración del entorno de desarrollo

La configuración del entorno, como siempre, la vamos a hacer con Maven. De esta forma resulta tan sencillo como añadir a nuestro pom.xml las siguientes líneas:

```
view plain print ?
01. ...
02. <repositories>
03. ...
04. <repository>
05. <id>repository.jboss.org</id>
06. <name>JBoss Maven Repository</name>
07. <url>http://repository.jboss.org/maven2</url>
08. <layout>default</layout>
09. </repository>
10. ...
11. </repositories>
12.
13. <dependencies>
14. ...
15. <dependency>
16. <groupId>org.hibernate</groupId>
17. <artifactId>hibernate</artifactId>
18. <version>3.2.6.ga</version>
19. </dependency>
20. <dependency>
21. <groupId>org.hibernate</groupId>
22. <artifactId>hibernate-search</artifactId>
23. <version>3.0.1.GA</version>
24. </dependency>
25. <dependency>
26. <groupId>org.hibernate</groupId>
27. <artifactId>hibernate-annotations</artifactId>
28. <version>3.3.1.GA</version>
29. </dependency>
30. ...
31. </dependencies>
32. ...
```

2008-06-13
Comercial - Ventas -
VALENCIA.

2008-06-12
T. Información - Analista
Funcional (otros) - MADRID.

2008-06-11
T. Información - Técnico de
Soporte (Help Desk) -
MALAGA.

2008-06-09
Otras Sin catalogar -
ASTURIAS.

2008-06-07
T. Información - Otros no
catalogados - MADRID.

Anuncios Google

4. Configuración de Hibernate Search

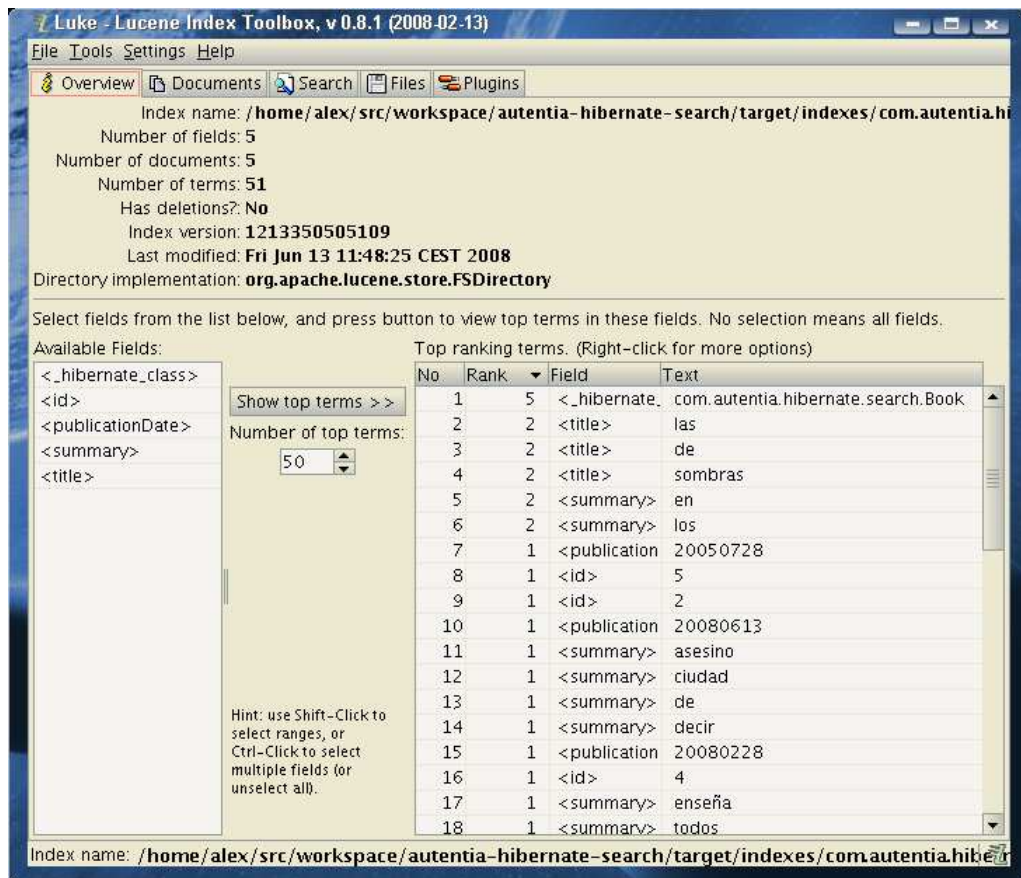
Hibernate Search ya viene configurado para poder trabajar en la mayoría de los casos. Por esto para empezar a trabajar tan sólo tendremos que añadir a nuestro hibernate.cfg.xml las siguientes líneas:

```
view plain print ?
01. <property name="hibernate.search.default.directory_provider">org.hibernate.search.store.RAMDirectoryProv
02. <!-- <property name="hibernate.search.default.directory_provider">org.hibernate.search.store.FSDirectoryI
03. <!-- <property name="hibernate.search.default.indexBase">/tmp/autentia-hibernate-search/indexes</property>
```

En la línea 1 le estoy diciendo que quiero que me guarde los índices en memoria. Es decir, al terminar la aplicación se perderán los índices. Esto no suele ser el comportamiento deseado, pero para hacer test automáticos nos va a venir muy bien :)

En la línea 2 y 3 os pongo comentado como configuramos Hibernate Search para que mantenga los índices en el disco duro. En concreto en la línea 3 es donde le decimos la ruta en el disco donde se guardarán los índices. Os recomiendo que sea una ruta fuera de vuestro servidor de aplicaciones, y deberíais hacer copias de seguridad de ella.

Si guardáis los índices en disco siempre podéis explorarlos y jugar con ellos, gracias a la herramienta Luke (<http://www.getopt.org/luke/>). Os dejo un pantallazo para que os hagáis una idea:



Por supuesto que hay un montón de cosas más que podemos configurar, tanto de Hibernate Search como de Lucene. Pero entrar en esto escapa de la intención de este tutorial.

Para saber más al respecto os recomiendo especialmente la documentación de Hibernate Search (http://www.hibernate.org/hib_docs/search/reference/en/html_single/) que es bastante clarificadora y es bastante corta (te la lees en un ratito). Y por supuesto la documentación de Luce para entender bien todos los conceptos, y exprimirlo al máximo (http://lucene.apache.org/java/2_3_2/).

5. Construyendo un POJO que sea "indexable"

Ahora vamos a crear un POJO que el sistema sea capaz de indexar. Usaremos anotaciones y nos queda algo tan sencillo como:

```
view plain print ?
01. @Entity
02. @Indexed
03. public class Book {
04.
05.     @Id
06.     @GeneratedValue
07.     @DocumentId
08.     private Integer id;
09.
10.     @Field
11.     private String title;
12.
13.     @Field
14.     private String summary;
15.
16.     @Field
17.     @DateBridge(resolution = Resolution.DAY)
18.     @Temporal(TemporalType.TIMESTAMP)
19.     private Date publicationDate;
20.
21.     ...
22.     // Resto de constructores, métodos, getter y setters, etc
23.     ...

```

Vamos a comentar exclusivamente las anotaciones de Hiberante Search:

- Línea 2, **@Indexed**: Esta anotación indica que este POJO debe ser indexado por Lucene cada vez que hagamos una operación con Hibernate. Es decir, al persistir, actualizar o borrar el POJO, Hibernate Search se encargará de mantener actualizado y sincronizado el índice de Lucene.
- Línea 7, **@DocumentId**: Indicamos que atributo será el identificador para la indexación. Esto es un tema de diseño de Hibernate Search, y es obligatorio.
- Línea 9, **@Field**: Hibernate Search sólo guardará en el índice los atributos que tengamos marcados con esta anotación. Por defecto esta anotación indexa el atributo "tokenizado" (es decir, separado en palabras), y no guarda el contenido del atributo en el índice. Esto último significa que para ver el contenido del atributo Hibernate Search tendrá que hacer una búsqueda en la base de datos (esto suele ser lo habitual y solo guardaremos el contenido en el índice si queremos hacer proyecciones).
- Línea 16, **@DateBridge**: Los índices de Lucene sólo guardan texto (por eso son búsquedas textuales ;), lo que hace Lucene es guardar la representación como texto de los atributos de nuestra clase. Todo esto es configurable, por ejemplo en el caso de las fechas, gracias a esta anotación, podemos decirle que parte de la fecha queremos indexar en el índice, por defecto

indexará toda la precisión del Date (hasta segundo y milisegundos), pero esto no es demasiado útil, así que le podemos marcar cuanto precisión queremos indexar. En el índice le estamos indicando que sólo guarde hasta el día, este incluido. Hay que tener en cuenta que para buscar estas fechas el texto de la consulta lo tendremos que expresar en el formato yyyyMMddHHmmssSSS, en tiempo GMT.

Os recuerdo que para saber más sobre estas anotaciones, sus parámetros, posibles configuraciones, ... lo mejor es acudir a la configuración de Hibernate Search.

Pues ya está señores, ya hemos terminado. Fácil ¿verdad? Ahora vamos a ver como podemos hacer algunas consultas.

6. Como hacer consultas en nuestros índices

Imaginemos que tenemos un Dao implementado con Spring (en los recursos podéis acceder a todo el código). Podríamos tener el siguiente método para hacer búsquedas textuales sobre una entidad concreta:

```
view plain print ?
01. @Transactional(readOnly = true)
02. public <T> List<T> findByFullText(Class<T> entityClass, String[] entityFields, String textToFind) {
03.     final FullTextSession fullTextSession = Search.createFullTextSession(getSession());
04.     final MultiFieldQueryParser parser = new MultiFieldQueryParser(entityFields, new StandardAnalyzer()
05.         org.apache.lucene.search.Query luceneQuery;
06.     try {
07.         luceneQuery = parser.parse(textToFind);
08.     } catch (ParseException e) {
09.         log.error("Cannot parse [" + textToFind + "] to a full text query", e);
10.         return new ArrayList<T>{0};
11.     }
12.     final Query query = fullTextSession.createFullTextQuery(luceneQuery, entityClass);
13.     return query.list();
14. }
```

- Línea 3: Creamos una sesión para hacer búsquedas textuales. Esta sesión se hace en base a la sesión de Hibernate.
- Línea 4 a 11: Preparamos la consulta de Lucene, en función al texto que nos han pasado y los campos que queremos inspeccionar.
- Línea 12: Lanzamos la consulta de Hibernate sobre la entidad que nos han dicho. No es obligatorio hacer consultas sobre entidades determinadas, podríamos lanzar consultar sobre cualquier entidad; pero es interesante, de cara al rendimiento, reducir las búsquedas siempre que sea posible, limitando las entidades sobre las que se tiene que lanzar la consulta.

Sobre las búsquedas, y en general aplicable para cualquier operación que hagamos con Hibernate, es muy recomendable hacerlas siempre en el marco de una transacción (el ejemplo lo conseguimos gracias a Spring con la anotación de la línea 1). De esta forma se optimiza el rendimiento.

Por ejemplo, si hacemos varias inserciones en la base de datos dentro de la misma transacción, el índice de Lucene sólo se actualizará con el commit de la transacción. Además, si algo sale mal, y se hacer rollback de la transacción, el índice no se llegará a tocar.

7. Recursos

Aquí os dejo todos los fuentes del ejemplo, donde podréis encontrar toda la configuración y los test completos donde podréis jugar haciendo diferentes búsquedas.

8. Conclusiones

Evidentemente hay mucho más de lo que hemos visto aquí. Pero sólo con lo que hemos tratado en este tutorial ya tenemos para hacer grandes cosas.

Además gracias a las anotaciones conseguimos desarrollar muy rápido, y el código es sencillo, limpio, y mantenible.

Espero que os haya gustado, a mi desde luego me parece una opción muy interesante.

9. Sobre el autor

Alejandro Pérez García, Ingeniero en Informática (especialidad de Ingeniería del Software)

Socio fundador de Autentia (Formación, Consultoría, Desarrollo de sistemas transaccionales)

<mailto:alejandropg@autentia.com>

Autentia Real Business Solutions S.L. - "Soporte a Desarrollo"

<http://www.autentia.com>

- Puedes opinar sobre este tutorial [haciendo clic aquí](#).
- Puedes firmar en nuestro libro de visitas [haciendo clic aquí](#).
- Puedes asociarte al grupo AdictosAlTrabajo en XING [haciendo clic aquí](#).
- Añadir a favoritos Technorati.



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Recuerda

Autentia te regala la mayoría del conocimiento aquí compartido ([Ver todos los tutoriales](#)). Somos expertos en: J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ... y muchas otras cosas.

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?, ¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos ...

Autentia = Soporte a Desarrollo & Formación.

info@autentia.com



Servicio de notificaciones:

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales.

Formulario de subscripción a novedades:

E-mail

Tutoriales recomendados

Nombre	Resumen	Fecha	Visitas	pdf
Primeros pasos con el motor de búsqueda Java Lucene	Os mostramos los primeros pasos con uno de los más extendidos motores de búsqueda dentro del mundo OpenSource y Java: También veremos lo sencillo que es monitorizar los índices con Luke e indexar PDFs con PDFBox	2006-07-31	7484	pdf
Proyecto con JSF Java Server Faces Myfaces, Maven y Eclipse: Hibernate (segunda parte)	En este artículo se va a continuar con el desarrollo de la aplicación Myfaces JSF con Maven multimódulo que comenzamos en un tutorial anterior. Además también se tratará de la integración de Hibernate con las aplicaciones.	2007-07-31	4210	pdf
Framework para indexación de documentos en Lucene: LIUS	En este tutorial vamos a ver como usar el framework LIUS (Lucene Index Update and Search) para indexar documentos en el motor de búsqueda textual Lucene	2007-10-23	2361	pdf
Introducción a Hibernate	Cesar Crespo nos enseña como utilizar unos de los sistemas más extendidos de mapeo de objetos a estructuras relacionales (tablas de base de datos)	2004-08-14	54569	pdf
Creación de una aplicación con Spring e Hibernate desde 0	Este tutorial vamos a explicar paso a paso cómo crear una pequeña aplicación usando Spring e Hibernate con anotaciones partiendo desde 0	2008-02-15	5210	pdf
Ficheros de mapeo de Hibernate desde las clases	Vamos a ver cómo las HibernateTools nos permiten generar de manera automática los ficheros de mapeo de Hibernate a partir de anotaciones en las clases	2008-06-02	549	pdf
Lucene: Analyzers, stemming y búsqueda de documentos similares.	En este tutorial vamos a ver cómo implementar un analizador semántico en nuestro idioma, potenciando la indexación y búsqueda, para terminar analizando la viabilidad de realizar búsquedas de documentos similares.	2008-02-22	1515	pdf
Spring + Hibernate + Anotaciones = Desarrollo Rápido en Java	Alejandro Pérez nos enseña lo fácil y rápido que es desarrollar en Java usando Spring e Hibernate, y usando anotaciones	2008-05-14	2147	pdf
Manejar dos bases de datos distintas con Hibernate	Alejandro Pérez nos enseña como manejar dos bases de datos distintas con Hibernate	2005-07-13	15687	pdf
Hibernate 3 y los tipos de datos para cadenas largas	En este tutorial se contará la experiencia que hemos tenido a la hora de manejar los diferentes tipos de datos existentes para grandes cadenas de texto, tales como el tipo Clob de Oracle, o los tipos TEXT de MySQL y SQLServer, utilizando la última versión	2007-03-23	7765	pdf

Nota:

Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento. Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores. En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo. Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.