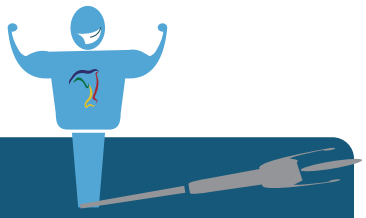
autentia

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...

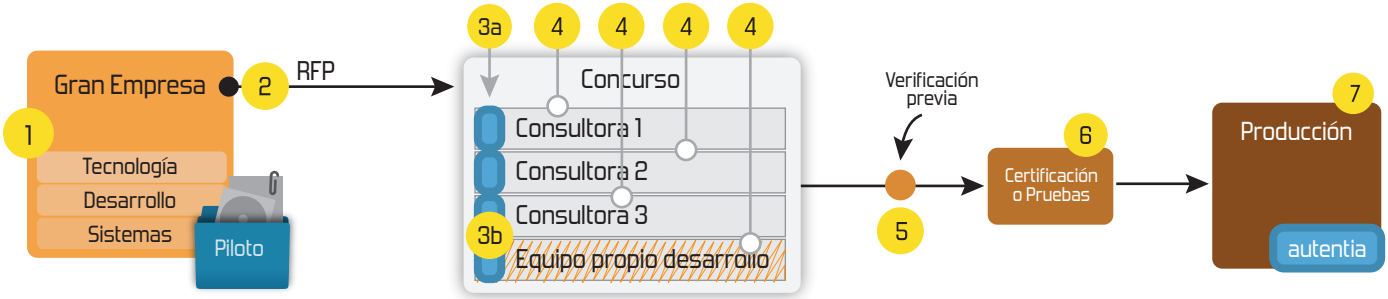
1. Desarrollo de componentes y proyectos a medida



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

Juan Diego Reyes Ponce

Consultor tecnológico de desarrollo de proyectos informáticos.

Ingeniero en Informática

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: 2014-09-24 Tutorial visitado 17 veces Descargar en PDF

Creación de un módulo AMP de Alfresco con arquetipo Maven

Índice de contenidos

1. Introducción

¿Qué es un módulo AMP de Alfresco?

Un módulo AMP (Alfresco Module Package) no es más que un conjunto de elementos que extienden la funcionalidad por defecto de Alfresco: contenidos, servicios, etc. Este fichero empaquetado (se puede asemejar a un archivo .jar) contiene los elementos necesarios para extender ciertas partes de la funcionalidades que proporciona nuestra instalación de Alfresco (depende de la versión ofrecerá unas funcionalidades u otras). Así pues, algunas de las funcionalidades que puede extender son:

- Definición de nuevos tipos de contenido
- Definición de aspectos (más información sobre aspectos aquí)
- Definición de propiedades o meta-información, que posteriormente podrá ser indexada para búsquedas en Lucene o Solr
- Generación de webscripts para añadir servicios personalizados en el catálogo de servicios de Alfresco: Ésta quizá sea la más potente de las opciones que nos ofrece los AMPs de Alfresco, ya que podremos hacer uso de su API para definir nuestros propios servicios (REST) para cubrir nuestras necesidades de negocio
- Personalización de elementos del IU de Alfresco: Definición de nuevas opciones en desplegables, definición de nuevos campos, adición de campos en formulario de búsqueda avanzada, estilos css, etc.

Para más información sobre AMPs (qué son, qué características tienen, qué estructura tienen, etc.), puedes consultar aquí

¿Cuándo tiene sentido construir un AMP?

Una de las razones de más peso de construir nuestro propio AMP es cuando necesitamos tener un módulo independiente con funcionalidad propia según los requisitos de negocio que tengamos. Dicho de otra forma, tendremos a nuestra disposición todas las propiedades, servicios, personalización de elementos de IU, conviviendo con el resto de funcionalidad por defecto que proporciona Alfresco, o incluso con otros AMPs de terceros.

Otra razón puede ser por ejemplo que necesitemos un nuevo tipo de contenido en Alfresco. Imaginamos que necesitamos un nuevo tipo de contenido que sea DocumentoDepartamento, que extiende del contenido por defecto de Alfresco, pero que incorpora el nombre del departamento y de la empresa al que pertenece el documento. Con la inclusión de este nuevo tipo en nuestro AMP, a la hora de crear o actualizar un documento, cuando se indique el tipo, se podrá elegir la opción de tipo DocumentoDepartamento, y automáticamente se incluirán nuestras nuevas propiedades. OJO --> y se indexarán sus valores para las futuras búsquedas.

Otra ventaja de definir nuestro propio AMP mediante aspectos por ejemplo, es que podremos aplicar dichos aspectos a los sitios (carpetas, ficheros) que necesitemos, de forma que para el resto de sitios será transparente el uso de estos aspectos, y sólo se aplicarán donde sea necesario.

Formas de generar un AMP

Existen varias formas de generar un módulo AMP de Alfresco. Podemos generar un AMP usando la herramienta de gestión de módulos, o bien a través de un arquetipo Maven. En este tutorial vamos a elegir la segunda opción, y veremos las dos vertientes que existen para generarlo

Últimos Tutoriales del Autor

» Construir un cliente REST con PowerBuilder .NET

» Spring BeanPostProcessor

2. Entorno

- Macbook pro core i7 con 16gb RAM
- SO: Mavericks
- IDE: Spring Tool Suite 3.4.0 RELEASE
- Apache Maven 3.1.1

3. Generación de AMP - AllInOne

En este apartado vamos a generar un AMP a partir del arquetipo maven, con la opción "AllInOne", es decir, vamos a generar un proyecto donde contengamos:

- Nuestro módulo AMP
- Una instalación embebida de Alfresco (v4.2e) -> Aplicaciones Alfresco, Share y Solr
- Un contenedor de servlets Apache Tomcat 7 embebido

Con esta solución tendremos todo un sistema de Alfresco corriendo en nuestra máquina, con el módulo AMP configurado y listo para ser usado. Veamos a continuación cómo conseguirlo.

Explicando el dominio del problema

Supongamos que nuestro cliente nos pide que necesita en los documentos que sube al gestor de contenidos Alfresco cierta información sobre el departamento al que pertenece el documento dentro de la empresa, por ejemplo:

- Identificador del departamento
- Nombre del departamento
- Nombre de la empresa cliente

Además, quiere una solución que permita que posteriormente mediante el desarrollo de un componente de búsqueda, esos campos sean susceptibles de ser buscados. Inicialmente podemos pensar en una solución donde todo se almacene en base de datos, pero es una solución muy costosa si el número de campos va creciendo (mayor número de índices de columna, búsquedas más costosas, etc.), por lo que tendremos que pensar en integrar dicha información con los propios documentos en Alfresco, beneficiándonos de la potencia de las búsquedas con Lucene.

Generando el proyecto Maven

Lo primero que vamos a hacer es generar la siguiente estructura de proyecto:

- -dpto_alfresco
- | pom.xml
  - - dpto_alfresco_amp
  - | pom.xml

Para ello, abrimos una consola y nos posicionamos en nuestro workspace. A continuación escribimos el siguiente comando:

```
mvn archetype:generate
```

Elegimos las opciones por defecto:

```
Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): 471:

[...]

1: 1.0-alpha-1
2: 1.0-alpha-2
3: 1.0-alpha-3
4: 1.0-alpha-4
5: 1.0
6: 1.1
Choose a number: 6:

[...]
```

Establecemos el 'groupId' y el 'artifactId' como 'com.autentia.tutoriales' y 'dpto_alfresco', respectivamente, y asignaremos para este ejemplo la v0.0.1 (Alfresco a veces tiene problemas con las versiones SNAPSHOT de los AMPs). Por último, la propiedad 'package' coincidirá con el 'groupId'

```
Define value for property 'groupId': : com.autentia.tutoriales
Define value for property 'artifactId': : dpto_alfresco
Define value for property 'version':  1.0-SNAPSHOT: : 0.0.1
Define value for property 'package':  com.autentia.tutoriales: :
Confirm properties configuration:
groupId: com.autentia.tutoriales
artifactId: dpto_alfresco
version: 0.0.1
package: com.autentia.tutoriales
```

Confirmamos y se nos generará un proyecto 'dpto_alfresco' en nuestro workspace.

Antes de continuar tendremos que modificar el tipo de proyecto como proyecto 'pom' (por defecto te asigna proyecto 'jar'). Por lo tanto, modificamos el packaging del pom.xml a 'pom':

```
<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/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.autentia.tutoriales</groupId>
  <artifactId>dpto_alfresco</artifactId>
  <version>0.0.1</version>
```

```
  <packaging>pom</packaging>

  <name>dpto_alfresco</name>
  <url>http://maven.apache.org</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Ahora es momento de generar nuestro módulo AMP 'AllInOne'. Nos posicionamos en nuestra recién creada carpeta 'dpto_alfresco' y escribimos:

```
mvn archetype:generate -DarchetypeCatalog=https://artifacts.alfresco.com/nexus/content/groups/public/archetype-catalog
```

Lo ejecutamos y nos aparece un menú con dos opciones:

```
Choose archetype:
    1: https://artifacts.alfresco.com/nexus/content/groups/public/archetype-catalog.xml -> org.alfresco.maven.archetyp
    2: https://artifacts.alfresco.com/nexus/content/groups/public/archetype-catalog.xml -> org.alfresco.maven.archetyp
Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): :
```

Éstos son los dos arquetipos que proporciona la integración de Maven con Alfresco 4.2e. Elegiremos la opción 2 : 'alfresco-allinone-archetype' (en el siguiente apartado veremos el ejemplo con la opción 1)

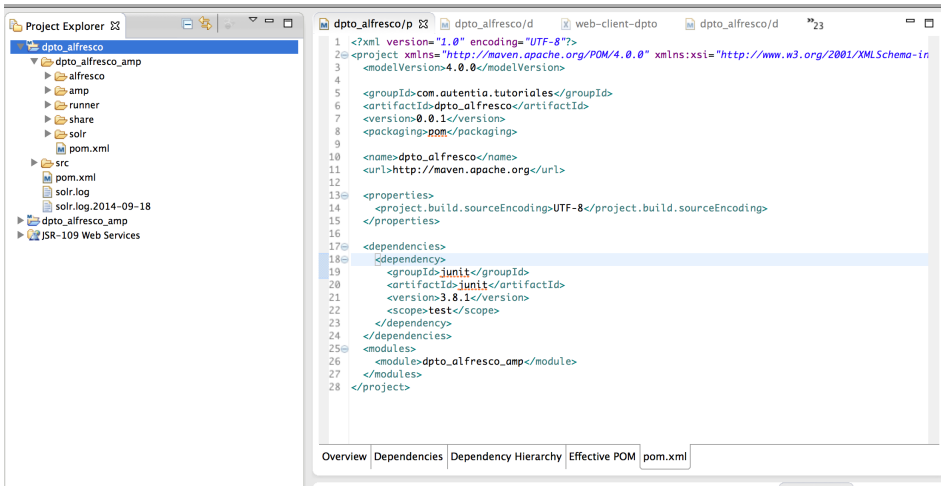
Asignaremos los valores por defecto, y aplicaremos los valores 'groupId'='com.autentia.tutoriales' y 'artifactId'='dpto_alfresco_amp'. NO confirmaremos los valores por defecto, ya que queremos modificar también la versión. Por tanto, le decimos que no a la confirmación y reestablecemos los valores 'groupId' y 'artifactId', y esta vez le asignamos la versión 0.0.1. El resto de valores los dejamos por defecto

```
Define value for property 'groupId': : com.autentia.tutoriales
Define value for property 'artifactId': : dpto_alfresco_amp
Define value for property 'version':  1.0-SNAPSHOT: : 0.0.1
Define value for property 'package':  (not used): :
Define value for property 'alfresco_target_groupId':  org.alfresco: :
Define value for property 'alfresco_target_version':  4.2.e: :
Confirm properties configuration:
    groupId: com.autentia.tutoriales
    artifactId: dpto_alfresco_amp
    version: 0.0.1
    package: (not used)
    alfresco_target_groupId: org.alfresco
    alfresco_target_version: 4.2.e
```

Si todo ha ido bien, nos generará un proyecto 'dpto_alfresco_amp' dentro de nuestro módulo padre 'dpto_alfresco', con la siguiente estructura:

- - dpto_alfresco_amp
- | alfresco -> Instalación de Alfresco 4.2e
- | amp -> Módulo AMP
- | runner -> Tomcat 7 embebido
- | share -> Instalación de aplicación share de Alfresco
- | solr -> Instalación de Apache Solr de Alfresco

Es momento de importarnos los proyectos a nuestro IDE. En este tutorial trabajaremos con Spring ToolSuite:



Construyendo el proyecto

Para construir el proyecto, nos posicionamos en el proyecto padre 'dpto_alfresco' y ejecutamos el siguiente comando:

```
mvn install
```

Llevará un tiempo la primera vez que lo hagamos, ya que se descargará todas las dependencias (instalación embebida de Alfresco, Share, Tomcat, etc.). Una vez que haya terminado tendremos instalado en nuestro repositorio de dependencias el AMP y el resto de proyectos.

Si ahora ejecutamos el proyecto, se desplegará una instalación de Alfresco 4.2e, la aplicación Share, Apache Solr y nuestro AMP (que por ahora no hace nada). Esta operación es costosa en memoria, por lo que modificaremos la memoria que necesita Maven para lanzar todo esto, ejecutando el siguiente comando:

```
export MAVEN_OPTS="-Xmx1024M -XX:MaxPermSize=512m"
```

A continuación, ejecutaremos:

```
mvn install -Prun
```

Cuando termine de ejecutar, entraremos en:

```
http://localhost:8080/alfresco
```

Y ya tendremos desplegado el proyecto. El usuario/pass será admin / admin

Creando el aspecto

Ya que el dominio del problema es la creación de atributos asociados a un documento con la información del departamento / empresa, tendremos que crear un aspecto que cubra éstas necesidades. Para ello:

1. Crearemos una carpeta 'models' dentro de /dpto_alfresco_amp/amp/src/main/amp/config/alfresco/module/amp/context/

2. Creamos un fichero 'dptoEmpresaModel.xml' que se encuentre en la carpeta 'models', y que contenga lo siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Modelo departamento y empresa -->

<model name="dpto:dptoEmpresaModel" xmlns="http://www.alfresco.org/model/dictionary/1.0">

   <!-- Optional meta-data about the model -->
    <description>Modelo departamento y empresa</description>
    <author>autentia</author>
    <version>1.0</version>

    <imports>
      <!-- Import Alfresco Dictionary Definitions -->
        <import uri="http://www.alfresco.org/model/dictionary/1.0" prefix="d"/>
    </imports>

    <namespaces>
        <namespace uri="dptoEmpresaModel.model" prefix="dpto"/>
    </namespaces>

    <aspects>
        <aspect name="dpto:dpto">
            <title>Departamento / Empresa</title>
            <properties>
                <property name="dpto:idDpto">
                    <title>ID de departamento</title>
                    <type>d:long</type>
                </property>

                <property name="dpto:nombreDpto">
                    <title>Nombre de departamento</title>
                    <type>d:text</type>
                </property>

                <property name="dpto:nombreEmpresa">
                    <title>Nombre de empresa</title>
                    <type>d:text</type>
                </property>
            </properties>
        </aspect>

    </aspects>

</model>
```

Aquí podemos observar varios elementos:

- Metadatos del modelo, como su nombre, descripción, autor y versión
- Import del diccionario de datos de Alfresco
- Namespace para evitar conflictos de nombres entre varios amps
- Definición del aspecto, que se compone de:
    - Título del aspecto
    - Propiedades del aspecto: Se definen las propiedades (en nuestro ejemplo sólo tenemos 3), con sus respectivos títulos y tipos de dato.

Necesitaremos a continuación modificar el contexto del servicio para referenciar a nuestro nuevo modelo. Modificaremos el archivo 'service-context.xml' con el siguiente contenido:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE beans PUBLIC '-//SPRING//DTD BEAN//EN' 'http://www.springframework.org/dtd/spring-beans.dtd'>
<!--
    Licensed to the Apache Software Foundation (ASF) under one or more
    contributor license agreements.  See the NOTICE file distributed with
```

```
    this work for additional information regarding copyright ownership.
    The ASF licenses this file to You under the Apache License, Version 2.0
    (the "License"); you may not use this file except in compliance with
    the License.  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.

-->
<beans>

    <bean id="es.autentia.tutoriales.dptoEmpresa.amp.dictionaryBootstrap"
        parent="dictionaryModelBootstrap"
        depends-on="dictionaryBootstrap">

        <property name="models">
            <list>
                <value>alfresco/module/${artifactId}/context/models/dptoEmpresaModel.xml</value>
            </list>
        </property>
    </bean>

</beans>
```

Con nuestro nuevo modelo referenciado, la próxima vez que arranquemos nuestro Alfresco embebido, tendremos instalado el aspecto. **Pero aún no será accesible, ya que necesitaremos modificar la IU para incluir el aspecto como una opción más a elegir**, entre otras funcionalidades.

Los módulos AMP ofrecen muchas posibilidades para modificar la IU de Alfresco, aunque en este tutorial se aplicarán en tres características de la interfaz:

- Selección del aspecto en reglas
- Mostrar las propiedades del aspecto en los detalles de los documentos
- Mostrar las propiedades del aspecto como atributos susceptibles de ser buscados en la búsqueda avanzada

Para ello crearemos una carpeta 'ui' colgando de la carpeta 'context' (al igual que la carpeta 'models'), y crearemos un archivo 'web-client-dpto-custom.xml', con el siguiente contenido:

```
    <alfresco-config>

    <!-- Lists the custom aspect in business rules Action wizard -->
    <!-- Definimos los aspectos creados para que sean accesibles en la aplicacion
        web para ser seleciconado como aspecto "Use the Custom Aspect in a Business
        Rule". -->
    <config evaluator="string-compare" condition="Action Wizards">
        <aspects>
            <aspect name="dpto:dpto" display-label="Departamento / Empresa" />
        </aspects>
    </config>

    <!-- Document Library config section -->
   <config evaluator="string-compare" condition="DocumentLibrary">

    <aspects>
         <!-- Aspects that a user can see -->
         <visible>
            <aspect name="dpto:dpto" display-label="Departamento / Empresa" />
         </visible>

         <!-- Aspects that a user can add. Same as "visible" if left empty -->
         <addable>
         </addable>

         <!-- Aspects that a user can remove. Same as "visible" if left empty -->
         <removeable>
         </removeable>
      </aspects>

   </config>


    <!-- Definimos la lista de propiedas para que sean visible en la vista del
        documento -->
    <config evaluator="aspect-name" condition="dpto:dpto">
        <property-sheet>
            <separator name="sepDptoEmpresa" display-label="Detalles Nodo - Departamento / Empresa"
                component-generator="HeaderSeparatorGenerator" />
            <show-property name="dpto:idDpto" />
            <show-property name="dpto:nombreDpto" />
            <show-property name="dpto:nombreEmpresa" />
        </property-sheet>
    </config>

    <!-- Búsqueda avanzada -->
    <config evaluator="string-compare" condition="Advanced Search">
        <advanced-search>
```

```
            <custom-properties>
                <meta-data aspect="dpto:dpto" property="dpto:idDpto" />
                <meta-data aspect="dpto:dpto" property="dpto:nombreDpto" />
                <meta-data aspect="dpto:dpto" property="dpto:nombreEmpresa" />
            </custom-properties>
        </advanced-search>
    </config>

</alfresco-config>
```

Tendremos que referenciar a continuación el archivo de personalización de IU desde el archivo 'module-context.xml' (OJO! no confundir con service-context), añadiendo el siguiente bean:

```
    <bean id="${groupId}.${artifactId}.ConfigBootstrap" class="org.alfresco.web.config.WebClientConfigBootstrap" init-meth
        <property name="configs">
         <list>
           <value>classpath:alfresco/module/${artifactId}/context/ui/web-client-dpto-custom.xml</value>
         </list>
        </property>
      </bean>
```

Una vez que lo tengamos todo, aplicaremos los cambios del AMP en el WAR de Alfresco. Para ello, tenemos que asegurarnos que en el pom.xml del módulo AMP tenemos bien referenciado el WAR cliente donde vamos a desplegar nuestro AMP (propiedad <alfresco.client.war>)

Nota importante: Es posible que la variable ${alfresco_target_amp_client_war} no esté bien definida en el pom, y genere un error al intentar incluir el amp en el war. Ante esto, será necesario definir esta propiedad en el propio pom de AMP con el valor 'alfresco' (si queremos desplegar el AMP en /alfresco) o 'share' (si queremos desplegar el AMP en /share)

Nos situaremos en la carpeta del proyecto AMP, y ejecutaremos el siguiente comando:

```
mvn clean integration-test -Pamp-to-war
```

Esto aplicará los cambios de nuestro AMP al WAR de Alfresco, y arrancará el contenedor de servlets.

(Si queréis limpiar por seguridad antes todas las carpetas de target, incluyendo los temporales, situaros en la carpeta dpto_alfresco_amp y escribid:)

```
mvn clean -Ppurge
```

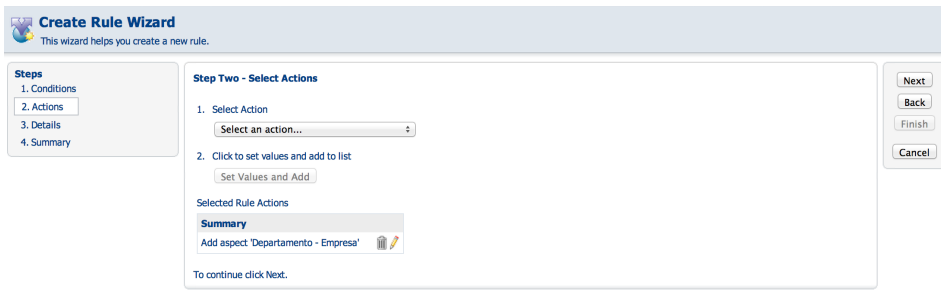
Si todo ha ido bien, entre los logs del arranque de Alfresco, podremos ver:

```
INFO  [repo.module.ModuleServiceImpl] [localhost-startStop-1] Found 1 module(s).
INFO  [repo.module.ModuleServiceImpl] [localhost-startStop-1] Installing module 'amp' version 0.0.1.
```

A continuación, escribimos:

```
http://localhost:8080/alfresco
```

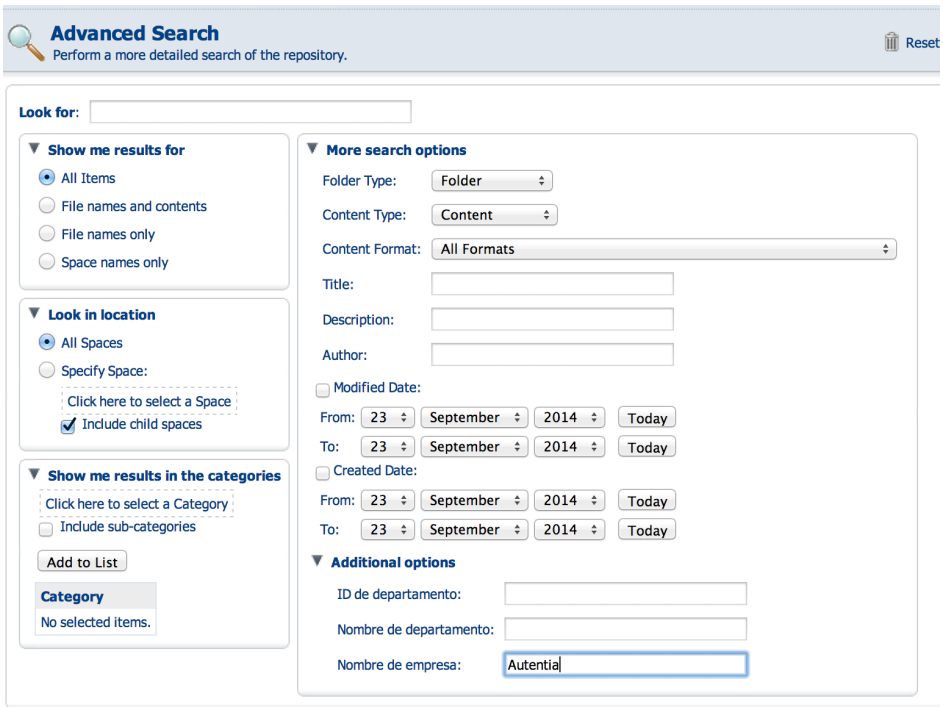
Entramos en la aplicación y vemos que los cambios se han aplicado. Para ello, podemos crear por ejemplo un nuevo sitio "DocsDpto", y crear una nueva regla de contenido, aplicando el aspecto recién creado a los documentos que se den de alta.



Damos a continuación de alta un nuevo documento dentro del espacio "DocsDpto" y vemos que nos pide los datos de ID Departamento, Nombre departamento y Nombre Empresa. Le damos los valores 1,"Departamento de Gestor de Contenidos Alfresco" y "Autentia", respectivamente.

Si entramos en los detalles del documento, veremos las propiedades del departamento en la sección 'Detalles Nodo - Departamento-Empresa'

Para probar la búsqueda avanzada, entramos en ella y probamos a buscar "Autentia" en el nombre de la empresa. Veremos que encuentra el contenido que acabamos de crear



4. Generación de AMP - StandAlone

La otra opción que nos ofrece el arquetipo Maven de Alfresco es la generación sólo del AMP, sin Alfresco embebido ni contenedor de servlets. Es una opción más ligera, pero que necesita de la instalación de un Alfresco. Además, la opción de AllInOne te permite probar de forma muy rápida los cambios en tu AMP, sin tener que redesplegar el AMP en Alfresco, por lo que recomendamos el uso de la solución AllInOne.

Los pasos a seguir son los mismos que en el apartado anterior, únicamente hay que elegir la opción 1 en la generación del proyecto mediante el arquetipo Maven de Alfresco

5. Conclusiones

Los módulos AMP de Alfresco nos sirven para extender de forma sencilla y en pocos pasos casi cualquier aspecto de una instalación de Alfresco: Extensión de tipos de contenido, modificación de la interfaz de usuario, etc.

Esto nos permite almacenar información adicional que más tarde podrá estar sujeta a las búsquedas en Alfresco (bien por servicios del API, o por búsquedas directas en Lucene o Solr). Hemos visto en el ejemplo del tutorial que hemos podido almacenar información adicional del departamento y la empresa a la que pertenece cada uno de los documentos.

Al igual que hemos definido un aspecto, podemos definir otro tipo de elementos (tipos, webscripts, etc.). Por ejemplo, con webscripts podremos definir servicios que añadan funcionalidad al catálogo de servicios que ofrece Alfresco, o bien definir un servicio que resuelva un problema de negocio en concreto (un ejemplo puede ser un tipo especial de búsqueda con Lucene, atacando al API de Lucene directamente).

6. Referencias

Podéis descargar el proyecto desde aquí: gitHub

Más referencias de interés:

- http://docs.alfresco.com/4.2/concepts/dev-extensions-modules-intro.html
- https://wiki.alfresco.com/wiki/Managing_Alfresco_Lifecyle_with_Maven
- http://www.flatironssolutions.com/blog/alfresco-maven-blog/

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

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

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