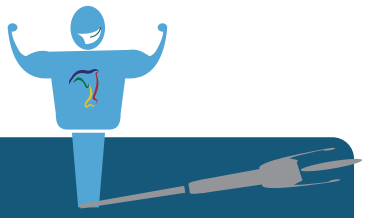
autentia

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...

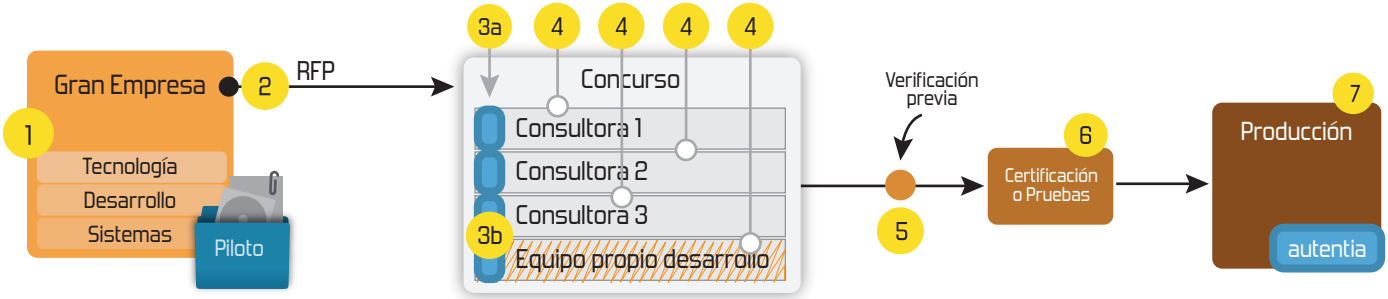
1. Desarrollo de componentes y proyectos a medida



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

adictos altrabajo

Hosting Patrocinado por enREDados.com

Home | Quienes Somos | Empleo | Foros | Tutoriales | Servicios Gratuitos | Contacte

Tutorial desarrollado por: Alejandro Perez García 2003-2005, nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.

Si te gusta lo que ves, **puedes contratarme** para impartir **cursos presenciales** en tu empresa o para ayudarte en proyectos (Madrid).

Contacta: alejandropg@autentia.com.

autentia real business solutions

Descargar este documento en formato PDF tomcatIIS.pdf

Mail server monitoring
Instant email & sms alerts.
Reports Full featured trial account.

Hospedaje Java
Hospedaje web con soporte de J2EE JSP Servlets EJB JTA JDBC JCA etc.

JSP Tutorials
JSP Made Easy With XMLSpy.
Syntax & Editing Help, Free D/L.

Iis Monitoring
Indepth monitoring, meet SLAs
Instant alerts. Free download now !

Anuncios Goooooogle Anúnciese en este sitio

Como configurar Tomcat + IIS (Internet Information Server de Microsoft)

Creación: 10-12-2005

Índice de contenidos

1. Introducción

2. Entorno

3. Precondiciones

4. El redirector

5. Configurando el redirector ISAPI

6. El fichero workers.properties

7. El fichero uriworkermap.properties

8. Añadir el directorio virtual al IIS

9. Añadir el filtro al IIS

10. Comprobar la instalación

11. Conclusiones

12. Sobre el autor

1. Introducción

Debido a que en Autentia (http://www.autentia.com) a menudo trabajamos con aplicaciones Web, nos encontramos con la necesidad de instalar un servidor de aplicaciones. La mayoría de servidores de aplicaciones también son capaces de servir recursos estáticos que se encuentran en el disco duro, como páginas html, imágenes gif, jpg, ... aunque normalmente esta opción no es la recomendada (los propios fabricantes suelen desaconsejarla ya que no se obtiene el rendimiento y la escalabilidad deseadas).

Los servidores de aplicaciones están optimizados para servir páginas dinámicas (Servlets, JSPs, ...) mientras que los servidores Web están optimizados para servir los recursos estáticos. Así que es habitual poner más de un servidor en cadena para mejorar el rendimiento y la escalabilidad de nuestras aplicaciones.

En este tutorial veremos como poner al frente un servidor Web como el IIS, que se encarga de servir las peticiones estáticas (es decir: las imágenes, los ficheros html, y en general todos los recursos que se encuentran en el disco duro), y a continuación montaremos un servidor capaz de servir componentes dinámicos. En nuestro caso un Tomcat, capaz de servir Servlets y páginas JSPs.

Vamos a ver como configurar el IIS para que sea él quien recibe todas las peticiones. El IIS determinará si el recurso solicitado lo puede servir el mismo o si debe pasarle la petición al Tomcat.

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portatil Ahtec Signal 259M MXM (Sonoma 2.1 GHz, 2048 MB RAM, 100 GB HD).
- Sistema Operativo: Microsoft Windows XP Profesional en Español
- IIS 5.1
- Tomcat 5.5.12

3. Precondiciones

Suponemos que ya tenemos correctamente instalados y en funcionamiento, tanto el IIS como el Tomcat.

En este tutorial sólo vamos a ver como hacer que trabajen conjuntamente.

Para ver como instalar cada servidor por separado buscar en otros tutoriales.

4. El redirector

Vamos a usar un plugin del IIS (filtro + extensión) como redirector entre el IIS y el Tomcat.

Este filtro comprueba la URL de la petición con una lista (que luego veremos como configurar). Si encuentra coincidencia le pasa la petición al Tomcat, y luego recoge la respuesta del Tomcat y la devuelve al navegador.

Si el filtro no encuentra coincidencia, será el IIS el que intente responder la petición.

Para instalar el redirector necesitamos tres piezas:

- **isapi_redirect.dll** - El filtro ISAPI para el IIS. La última versión (la 1.2.15) la podemos descargar de http://apache.org.es/tomcat/tomcat-connectors/jk/binaries/win32/jk-1.2.15/
- **workers.properties** - El fichero donde se describe el host y el puerto donde están los “workers”. Estos workers son lo distintos Tomcat que están atendiendo las peticiones (tendrán que estar definidos los conectores correspondientes en el fichero server.xml del Tomcat). Podemos encontrar un ejemplo de este fichero en el directorio “conf” de los fuentes del redirector http://apache.org.es/tomcat/tomcat-connectors/jk/source/jk-1.2.15/ (durante el proceso de configuración veremos nuestro propio ejemplo).
- **uriworkermap.properties** - El fichero donde se define la relación entre la URL y el worker que tiene que atender la petición. Podemos encontrar un ejemplo de este fichero en el directorio “conf” de los fuentes del redirector http://apache.org.es/tomcat/tomcat-connectors/jk/source/jk-1.2.15/ (durante el proceso de configuración veremos nuestro propio ejemplo).

5. Configurando el redirector ISAPI

Suponemos que hemos copiado isapi_redirect.dll en %TOMCAT_HOME%\bin\win32\i386. Donde %TOMCAT_HOME% es el directorio donde tenemos instalado el Tomcat (en mi caso c:\java\apache-tomcat-5.5.12).

Para configurarlo tenemos dos opciones, modificar el registro de Windows o crear un fichero de propiedades junto al fichero isapi_redirect.dll. Yo he elegido la opción del fichero de propiedades porque pienso que es menos intrusiva y porque permite tener varios redirectores con configuraciones diferentes.

El fichero de propiedades se tiene que llamar igual que la dll del redirector, con extensión “.properties”. En nuestro caso como la dll se llama “isapi_redirect.dll”, el fichero de configuración se llamará “isapi_redirect.properties”.

A continuación se muestra un ejemplo del fichero de propiedades:

```
# Configuration file for the Jakarta ISAPI Redirector

# The path to the ISAPI Redirector Extension, relative to the website
# This must be in a virtual directory with execute privileges
extension_uri=/jakarta/isapi_redirect.dll

# Full path to the log file for the ISAPI Redirector
log_file=c:\java\apache-tomcat-5.5.12\logs\isapi_redirect.log

# Log level (debug, info, warn, error or trace)
log_level=info
```

```
# Full path to the workers.properties file
worker_file=c:\java\apache-tomcat-5.5.12\conf\workers.properties

# Full path to the uriworkermap.properties file
worker_mount_file=c:\java\apache-tomcat-5.5.12\conf\uriworkermap.properties
```

En el ejemplo se aprecia como hago referencia a mi instalación del Tomcat (c:\java\apache-tomcat-5.5.12), vosotros tendréis que cambiar estos valores por los correctos para vuestra instalación. También se puede apreciar como hacemos referencia al fichero de “workers” y el de URLs que mencionábamos en el punto anterior. Más tarde veremos el contenido de estos dos ficheros.

Con el atributo extension_uri estamos haciendo referencia a un directorio virtual del IIS. Más adelante veremos como configurar este directorio virutal.

6. El fichero workers.properties

Este fichero lo copiamos en la localización que hemos indicado en el atributo “worker_file” del fichero de propiedades “isapi_redirect.properties”. En mi caso c:\java\apache-tomcat-5.5.12\conf\workers.properties.

A continuación mostramos un ejemplo de este fichero:

```
# workers.properties -
#
# This file provides jk derived plugins with the needed information to
# connect to the different tomcat workers.  Note that the distributed
# version of this file requires modification before it is usable by a
# plugin.
#
# As a general note, the characters $( and ) are used internally to define
# macros. Do not use them in your own configuration!!!
#
# Whenever you see a set of lines such as:
# x=value
# y=$(x)\something
#
# the final value for y will be value\something
#
# Normaly all you will need to do is un-comment and modify the first three
# properties, i.e. workers.tomcat_home, workers.java_home and ps.
# Most of the configuration is derived from these.
#
# When you are done updating workers.tomcat_home, workers.java_home and ps
# you should have 3 workers configured:
#
# - An ajp12 worker that connects to localhost:8007
# - An ajp13 worker that connects to localhost:8009
# - A jni inprocess worker.
# - A load balancer worker
#
# However by default the plugins will only use the ajp12 worker. To have
# the plugins use other workers you should modify the worker.list property.
#
#

# OPTIONS ( very important for jni mode )

#
# workers.tomcat_home should point to the location where you
# installed tomcat. This is where you have your conf, webapps and lib
# directories.
#
workers.tomcat_home=C:\java\apache-tomcat-5.5.12

#
# workers.java_home should point to your Java installation. Normally
# you should have a bin and lib directories beneath it.
#
workers.java_home=C:\java\jdk1.5.0_06

#
# You should configure your environment slash... ps=\ on NT and / on UNIX
# and maybe something different elsewhere.
#
ps=\

#
#------ ADVANCED MODE ----------------------------------------------
#-------------------------------------------------------------------
#

#
#------ DEFAULT worket list ----------------------------------------
#-------------------------------------------------------------------
#
#
# The workers that your plugins should create and work with
#
# Add 'inprocess' if you want JNI connector
worker.list=loadbalancer
# , inprocess
```

```
#
#------ DEFAULT worker1 WORKER DEFINITION ----------------------------
#---------------------------------------------------------------------
#

worker.worker1.port=8007
worker.worker1.host=localhost
worker.worker1.type=ajp13
#
# Specifies the load balance factor when used with
# a load balancing worker.
# Note:
#  ----> lbfactor must be > 0
#  ----> Low lbfactor means less work done by the worker.
worker.worker1.lbfactor=1

#
#------ DEFAULT worker2 WORKER DEFINITION ----------------------------
#---------------------------------------------------------------------
#

worker.worker2.port=8009
worker.worker2.host=localhost
worker.worker2.type=ajp13
#
# Specifies the load balance factor when used with
# a load balancing worker.
# Note:
#  ----> lbfactor must be > 0
#  ----> Low lbfactor means less work done by the worker.
worker.worker2.lbfactor=1

#
# Specify the size of the open connection cache.
#worker.ajp13.cachesize

#
#------ DEFAULT LOAD BALANCER WORKER DEFINITION ----------------------
#---------------------------------------------------------------------
#

#
# The loadbalancer (type lb) workers perform wighted round-robin
# load balancing with sticky sessions.
# Note:
#  ----> If a worker dies, the load balancer will check its state
#        once in a while. Until then all work is redirected to peer
#        workers.
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=worker1, worker2


#
#------ DEFAULT JNI WORKER DEFINITION--------------------------------
#---------------------------------------------------------------------
#

#
# Defining a worker named inprocess and of type jni
# Note that the name and the type do not have to match.
#
worker.inprocess.type=jni

#
#------ CLASSPATH DEFINITION -----------------------------------------
#---------------------------------------------------------------------
#

#
# Additional class path components.
#
worker.inprocess.class_path=$(workers.tomcat_home)$(ps)lib$(ps)tomcat.jar

#
# Setting the command line for tomcat.
# Note: The cmd_line string may not contain spaces.
#
worker.inprocess.cmd_line=start

# Not needed, but can be customized.
#worker.inprocess.cmd_line=-config
#worker.inprocess.cmd_line=$(workers.tomcat_home)$(ps)conf$(ps)server.xml
#worker.inprocess.cmd_line=-home
#worker.inprocess.cmd_line=$(workers.tomcat_home)

#
# The JVM that we are about to use
#
# This is for Java2
#
# Windows
worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)bin$(ps)classic$(ps)jvm.dll
# IBM JDK1.3
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)bin$(ps)classic$(ps)libjvm.so
# Unix - Sun VM or blackdown
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)lib$(ps)i386$(ps)classic$(ps)libjvm.so
```

```
#
# And this is for jdk1.1.X
#
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)bin$(ps)javai.dll


#
# Setting the place for the stdout and stderr of tomcat
#
worker.inprocess.stdout=$(workers.tomcat_home)$(ps)logs$(ps)inprocess.stdout
worker.inprocess.stderr=$(workers.tomcat_home)$(ps)logs$(ps)inprocess.stderr

#
# Setting the tomcat.home Java property
#
#worker.inprocess.sysprops=tomcat.home=$(workers.tomcat_home)

#
# Java system properties
#
# worker.inprocess.sysprops=java.compiler=NONE
# worker.inprocess.sysprops=myprop=mypropvalue

#
# Additional path components.
#
# worker.inprocess.ld_path=d:$(ps)SQLLIB$(ps)bin
#
```

En el ejemplo se ve como con los atributos “workers.tomcat_home” y “workers.java_home” ajustamos el fichero a nuestra entorno (directorio de instalación del tomcat y java).

Luego con “worker.list” definimos la lista de procesos que atenderán las peticiones. En el ejemplo definimos un único worker, balanceador de carga, que configuramos más adelante en el mismo fichero.

A continuación definimos cada uno de los workers, indicando puerto, host y tipo (ajp12 está obsoleto y sólo está disponible para Tomcat 3.2.x y 3.3.x). Estos workers deben estar dados de alta en el fichero server.xml del Tomcat. Según nuestro ejemplo deberíamos tener definidos dentro de <Service name="Catalina"> los siguientes conectores:

```
<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8007"
        enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />

<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8009"
        enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />
```

A continuación definimos un worker que será el balanceador de carga (“worker.loadbalancer.type”). Este es el worker que especificamos en el “worker.list”. También indicamos los workers que se utilizarán para hacer el balanceo (“worker.loadbalancer.balanced_workers”). Este balanceador de carga reparte peticiones por round-robin, es decir una petición a cada worker de manera consecutiva.

El resto del fichero no haría falta tocarlo.

7. El fichero uriworkermap.properties

Este fichero lo copiamos en la localización que hemos indicado en el atributo “worker_mount_file” del fichero de propiedades “isapi_redirect.properties”. En mi caso c:\java\apache-tomcat-5.5.12\conf\workers.properties.

A continuación mostramos un ejemplo de este fichero:

```
# uriworkermap.properties - IIS
#
# The general syntax for this file is:
# [URL]=[Worker name]

/admin/*=loadbalancer
/manager/*=loadbalancer
/jsp-examples/*=loadbalancer
/servlets-examples/*=loadbalancer

# Optionally filter out all .jpeg files inside that context
# For no mapping the url has to start with exclamation (!)

!/jsp-examples/*.jpeg=loadbalancer
!/jsp-examples/*.gif=loadbalancer
```

Nótese como en este fichero se define la URL y se le asigna el worker que debe servirla. En el ejemplo hemos elegido que todas las peticiones sean atendidas por el balanceador de carga, pero podríamos elegir que cada petición se atendiera por un worker distinto.

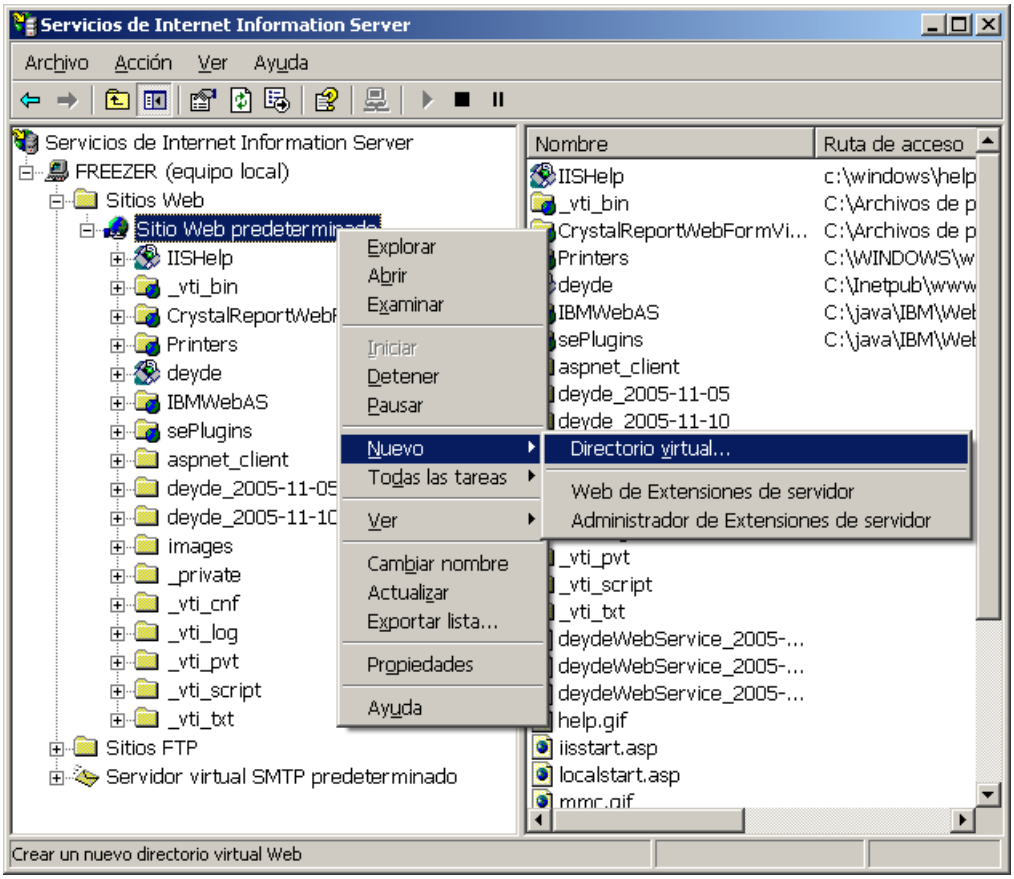
Nótese como se pueden poner exclusiones. Las dos últimas líneas del fichero, que empiezan por "!", están indicando que el Tomcat no debe servir ni los ficheros ".jpeg" ni los ".gif" dentro de jsp-examples. Esto va a provocar que las páginas se vean sin imágenes, a no ser que configuremos un directorio virtual en el IIS, para que sea este el que sirva las imágenes (en nuestro ejemplo deberíamos crear un directorio virtual apuntando a c:\java\apache-tomcat-5.5.12\webapps\jsp-examples).

8. Añadir el directorio virtual al IIS

Ahora necesitamos añadir un nuevo directorio virtual en el IIS. Este directorio virtual es el que hacíamos referencia con el atributo "extension_uri" del fichero "isapi_redirect.properties".

Abrimos la consola de administración del IIS (Inicio --> Panel de Control --> Herramientas administrativas --> Servicios de Internet Information Server).

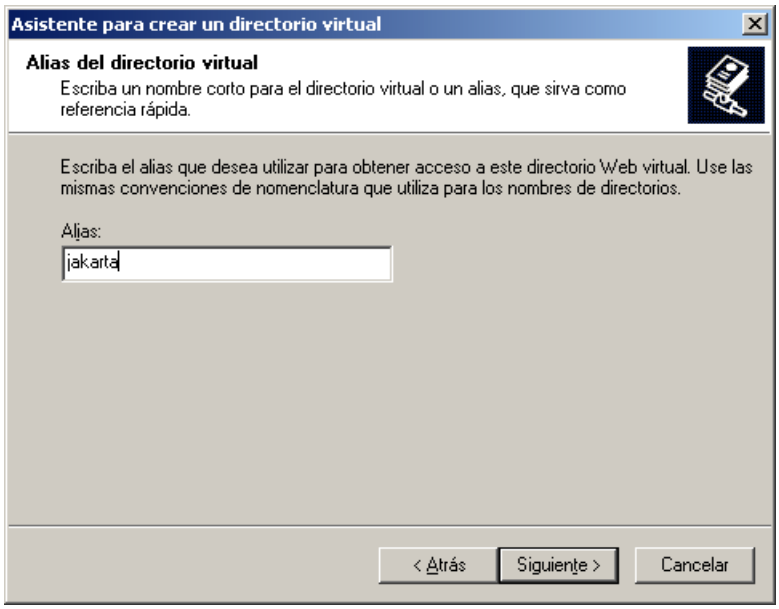
Para añadir el directorio virtual pinchamos botón derecho sobre Sitio Web predeterminado --> Nuevo --> Directorio virtual ...



Nos aparece el asistente para añadir un nuevo directorio virtual:

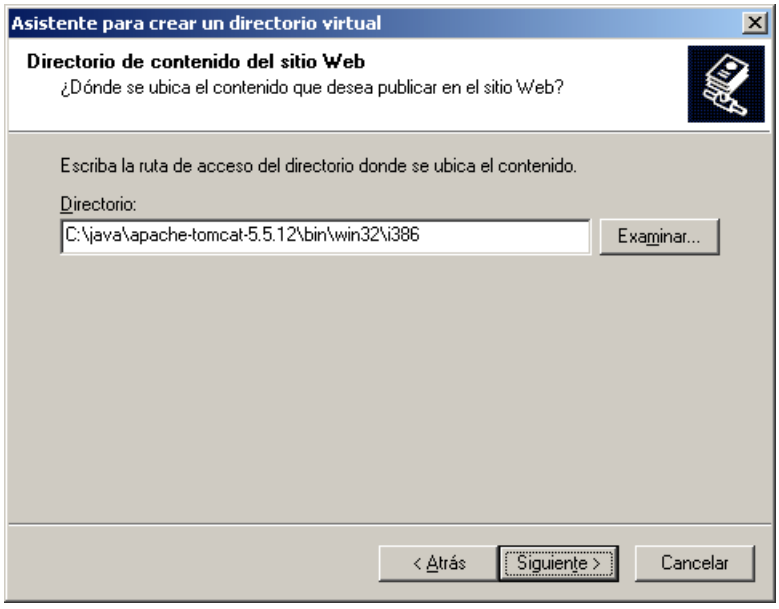


Pulsamos “Siguiente >”.



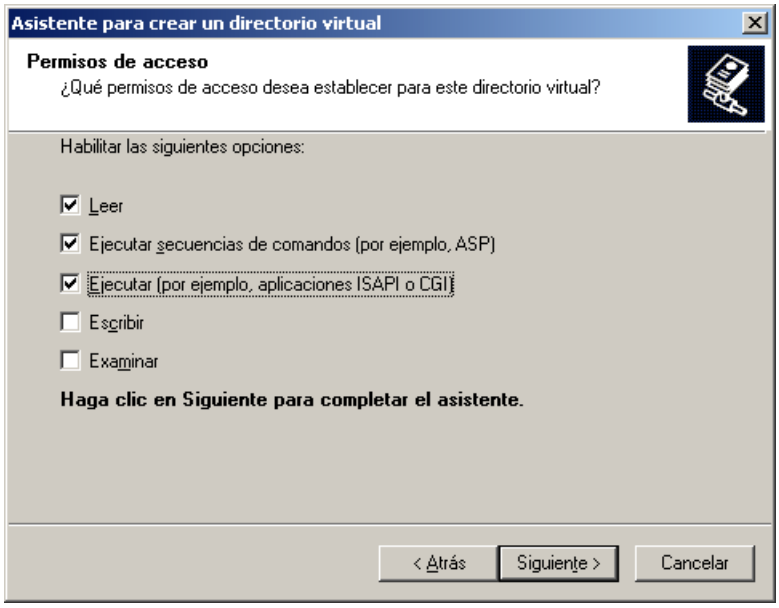
Escribimos “jakarta” como alias del virtual host. Este alias debe coincidir con lo que pusimos en el atributo “extension_uri” del fichero de propiedades.

Pulsamos “Siguiente >”.



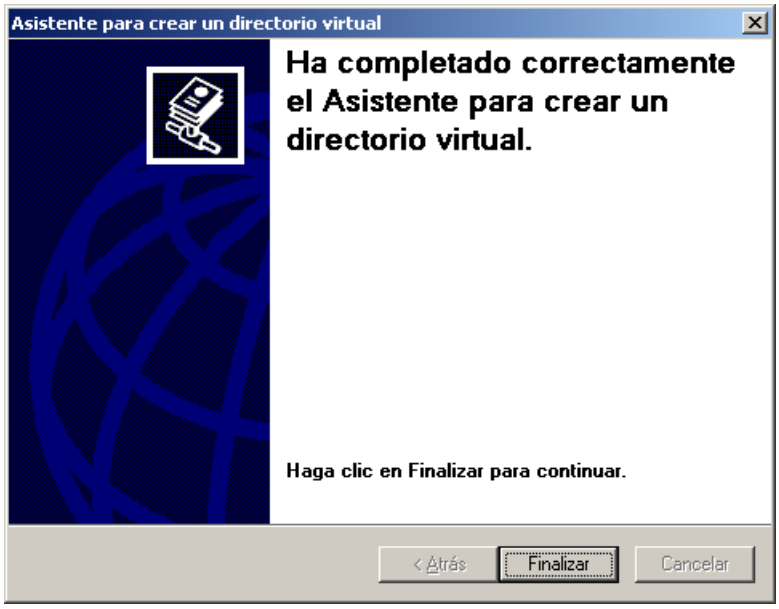
Ponemos la ruta física al directorio donde habíamos copiado el fichero “isapi_redirect.dll”. En mi caso c:\java\apache-tomcat-5.5.12 \bin\win32\i386.

Pulsamos “Siguiente >”.



Le damos permisos de ejecución.

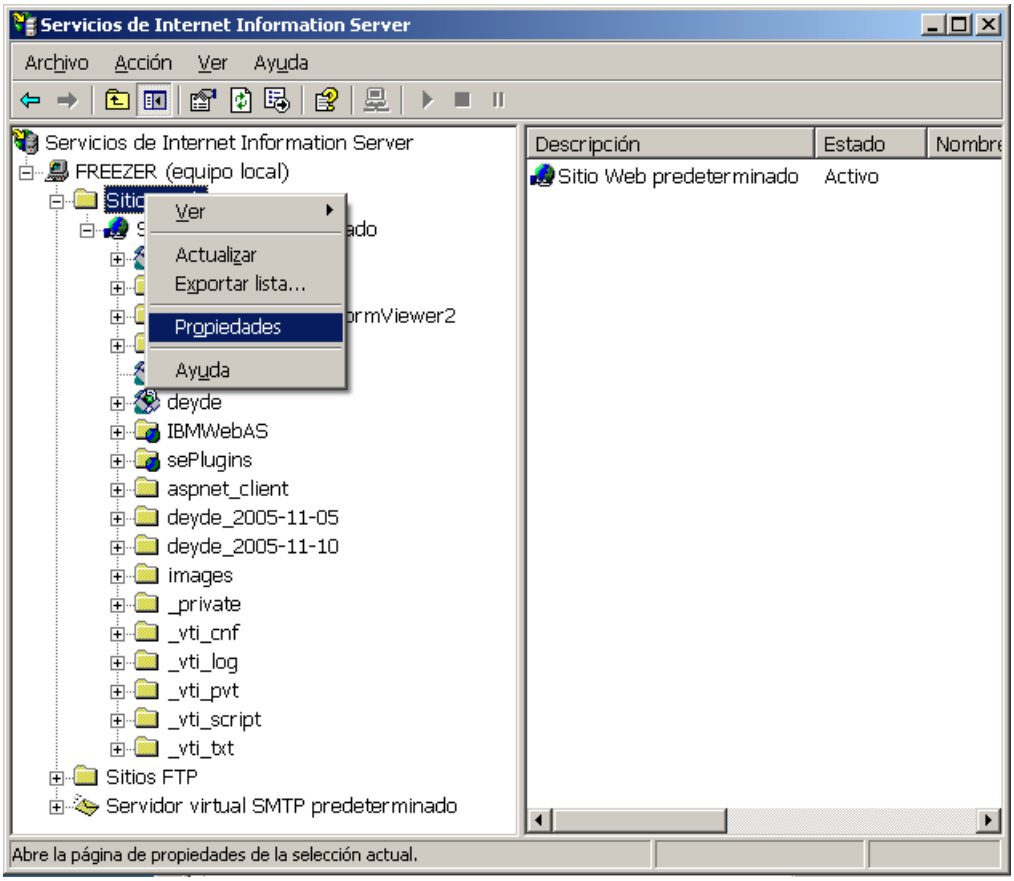
Pulsamos “Siguiente >”.

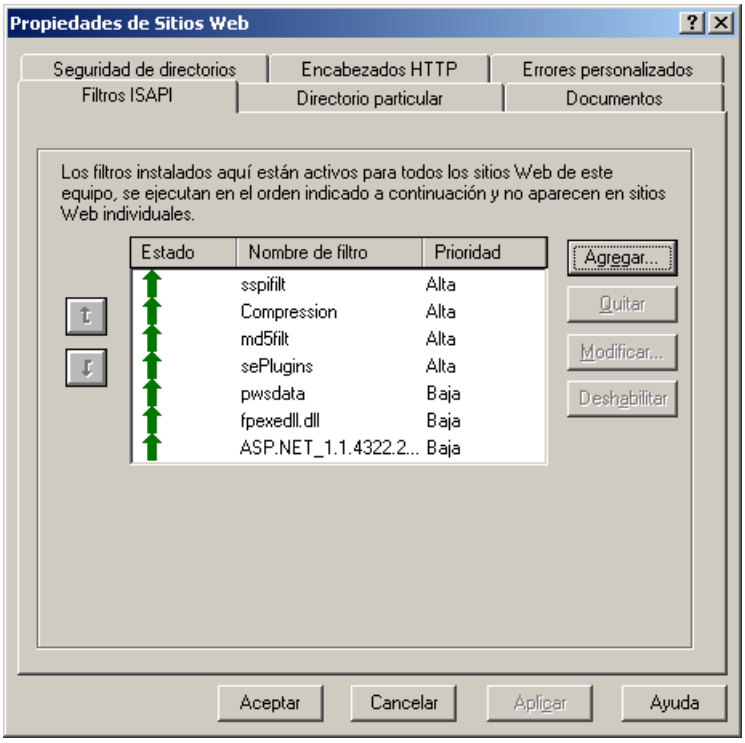


Pulsamos “Finalizar” para terminar el asistente.

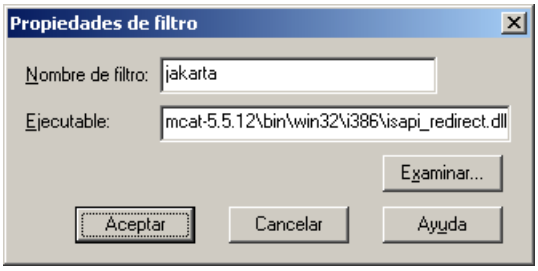
9. Añadir el filtro al IIS

Ahora necesitamos añadir el filtro “isapi_redirect.dll” al IIS. Para ello, pinchamos botón derecho sobre Sitios Web --> Propiedades.



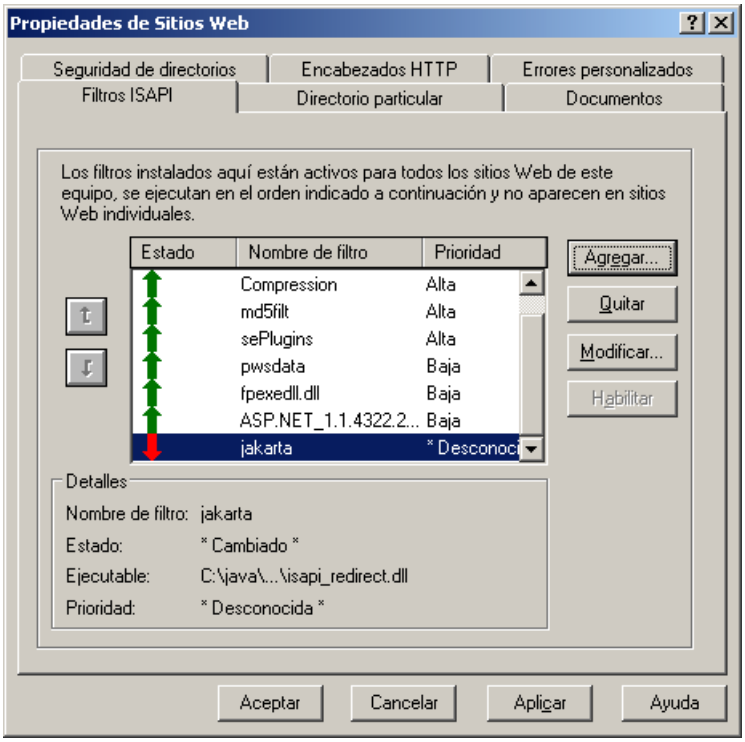


En la pestaña de “Filtros ISAPI” pinchamos sobre el botón “Agregar ...”.

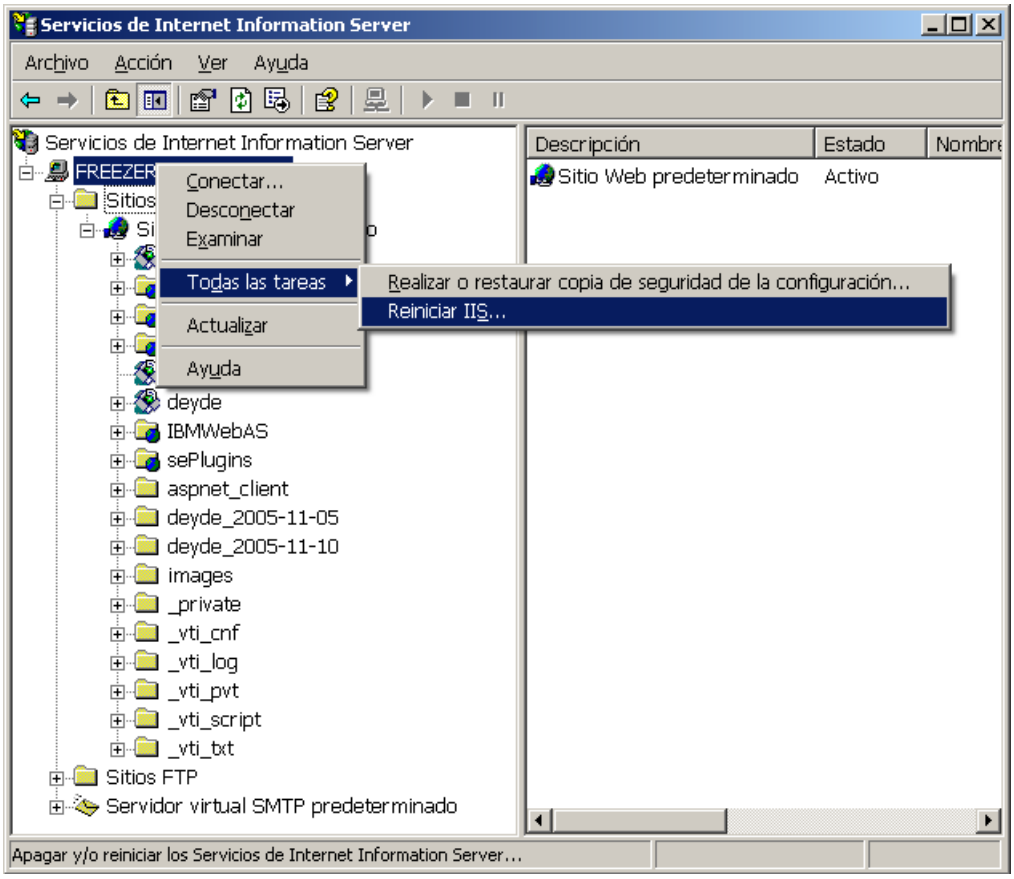


Ponemos un nombre para el filtro que sea significativo, yo he puesto “jakarta”, e indicamos la ruta del ejecutable, es decir la ruta donde se encuentra el fichero “isapi_redirect.dll”.

Pinchamos sobre “Aceptar”.

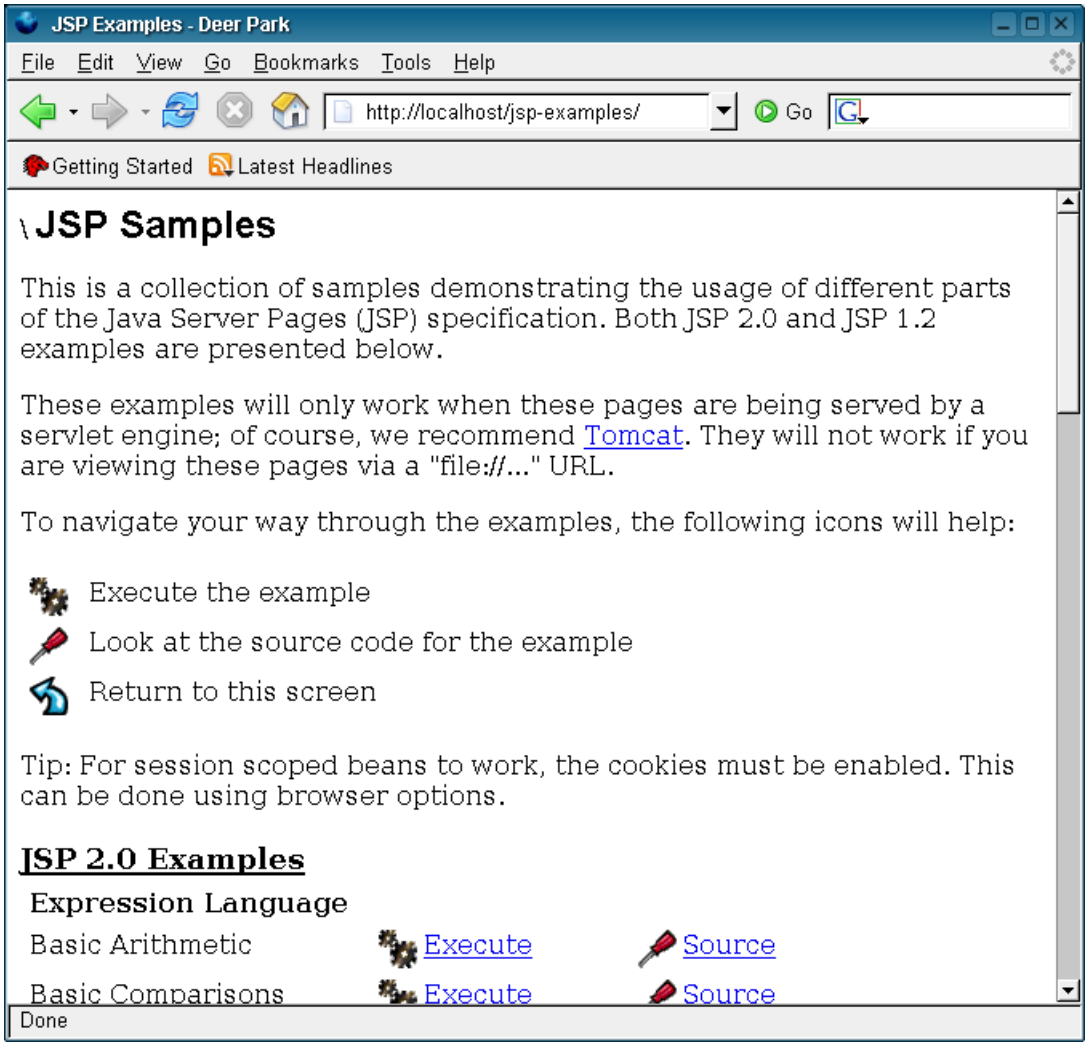


Vemos que el filtro está añadido pero no está levantado. Para ello necesitamos reiniciar el IIS. Para ello pinchamos botón derecho sobre el equipo local --> Todas las tareas --> Reiniciar IIS.



10. Comprobar la instalación

Podemos probar que todo ha ido correctamente abriendo nuestro navegador y escribiendo la URL http://localhost/jsp-examples/



Se puede ver como en la URL no hemos especificado el puerto 8080 del Tomcat. Es decir estamos haciendo la petición por el puerto 80 a través del IIS.

11. Conclusiones

Aunque en un principio puede parecer un poco complicado, merece la pena el esfuerzo, ya que las distintas configuraciones nos pueden dar mucho juego.

Podemos montan un IIS que reciba todas las peticiones. Con esto conseguimos que todas las peticiones entren por el puerto 80, con independencia de si las sirve el IIS o el Tomcat. Esto hace que no sea necesario abrir puertos adicionales en nuestros firewalls ni en los de nuestros clientes.

Además jugando con los workers podemos tener un IIS que reparta el trabajo ha distintos Tomcats, que pueden estar incluso en distintas máquinas. Eso puede mejorar el rendimiento de nuestras aplicaciones, y mejora la escalabilidad del sistema.

12. Sobre el autor

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

Dir. Implantación y Rendimiento

Formador en tecnologías J2EE, ADOO, UML

mailto:alejandropg@autentia.com

Autentia Real Business Solutions S.L.

http://www.autentia.com

Si desea contratar formación, consultoria o desarrollo de piezas a medida puede contactar con

soluciones reales para su negocio

Autentia S.L. Somos expertos en:
J2EE, C++, OOP, UML, Vignette, Creatividad ..
y muchas otras cosas

Nuevo servicio de notificaciones

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales, inserta tu dirección de correo en el siguiente formulario.

Subscribirse a Novedades	
e-mail	
	Enviar

Otros Tutoriales Recomendados (También ver todos)

Nombre Corto	Descripción
Instrumentación de componentes Java usando JMX	Os mostramos como utilizar las caracteristicas avanzadas de los servidores de aplicaciones Java/J2EE para realizar la instrumentación de componentes Java usando JMX y poniendo como ejemplo la gestión dinamica de propiedades entre nodos
Certificados en IIS para activación SSL	En este tutorial vamos a habilitar el soporte SSL (Secure Socket Layer, comunicación segura por https) en un servidor IIS (Internet Information Server de Microsoft).
Invocación a un componente COM desde Java	Os mostramos como invocar a un servidor de automatización OLE (COM) desde una aplicación Java, a través del producto gratuito JACOB
Patrones de diseño J2EE	Os mostramos una interpretación particular de los patrones de diseño J2EE
Evitar doble-click en JSPs y Struts	Os mostramos como construir unas librerías de TAGs para evitar el problema de doble-click en aplicaciones JSP y como se soluciona (qué teneis que hacer) con el framework Struts
Web Services en tu IPAQ	Cesar Crespo nos enseña como programar accesos Web Services desde tu IPAQ en Visual C++ con PocketSOAP, Apache SOAP y Axis
JSF y comparativa con Struts	Os mostramos los pasos necesarios para empezar a utilizar JSF (Java Server Faces) y su comparación / relación con Struts
Desarrollo Struts con XDoclet	Alejandro Perez nos enseña como simplificar el desarrollo de aplicaciones J2EE basadas en Struts, automatizando la generación de código con XDoclet
Activar el soporte SSL en Struts	Os mostramos las particularidades de uso y configuración de Struts para trabajar con SSL
Informes en Java con iReports	Cristhian Herrera, desde Ecuador, nos enseña como instalar y utilizar el iReports para la construcción de informes en tecnología Java.

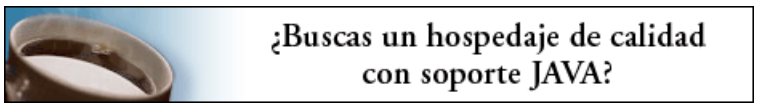
Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

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.

Patrocinados por enredados.com .... Hosting en Castellano con soporte Java/J2EE



www.AdictosAlTrabajo.com Opimizado 800X600