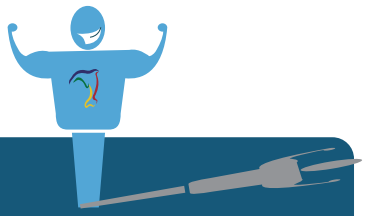
autentia

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...

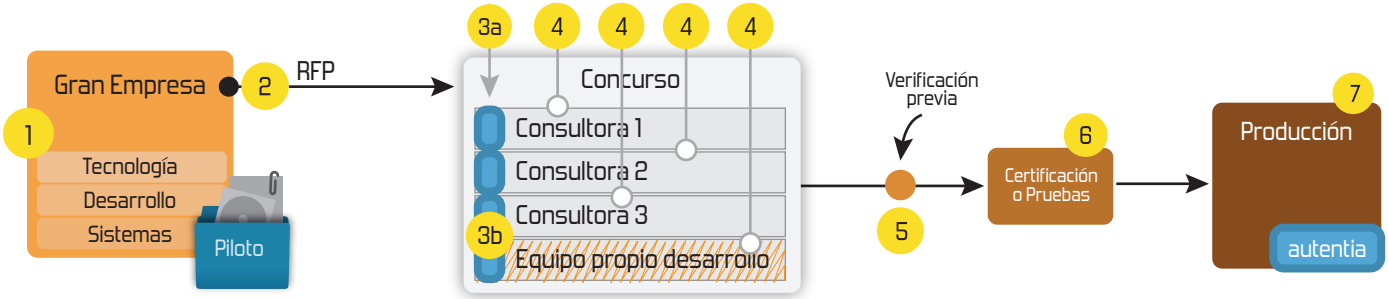
1. Desarrollo de componentes y proyectos a medida



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 Powered by autentia
Hosting Patrocinado por enREDados.com enREDados

Lanzado TNTConcept versión 0.6 (12/07/2007)

Desde Autentia ponemos a vuestra disposición el software que hemos construido (100% gratuito y sin restricciones funcionales) para nuestra gestión interna, llamado TNTConcept (au**T**e**NT**ia).

Construida con las últimas tecnologías de desarrollo Java/J2EE (Spring, JSF, Acegi, Hibernate, Maven, Subversion, etc.) y disponible en licencia GPL, seguro que a muchos profesionales independientes y PYMES os ayudará a organizar mejor vuestra operativa.

Las cosas grandes empiezan siendo algo pequeño ..... Saber más en: http://tntconcept.sourceforge.net/

Contacte con Cristóbal González
criskerberos-tutoriales@yahoo.com

Autor: Cristóbal González Almirón es consultor de desarrollo de proyectos informáticos.

Su experiencia profesional se ha desarrollado en empresas como Compaq, HP, Mapfre, Endesa, Repsol, Universidad Autónoma de Madrid, en las áreas de Desarrollo de Software (Orientado a Objetos), tecnologías de Internet, Técnica de Sistemas de alta disponibilidad y formación a usuarios.

NUEVO CATÁLOGO DE SERVICIOS DE AUTENTIA (PDF 6,2MB)

www.adictosaltrabajo.com es el Web de difusión de conocimiento de www.autentia.com

autentia real business solutions

Catálogo de cursos

Descargar este documento en formato PDF jsfmaven.pdf

Firma en nuestro libro de Visitas <-----> **Asociarme al grupo AdictosAlTrabajo en eConozco**

www.trabajaportucuenta.com Comentarios: anuncios Google

Fecha de creación del tutorial: 2007-05-28

Proyecto con JSF Java Server Faces Myfaces, Maven y Eclipse

Proyecto con JSF Java Server Faces Myfaces, Maven y Eclipse

Resumen de lo que vamos a ver en el tutorial

Introducción

Requisitos

Instalación de Maven

Creación de un proyecto maven 2 para MyFaces

Instalación del arquetipo de myfaces

Creando el proyecto con el arquetipo de maven

Compilando la aplicación

Probando la aplicación

Integrando nuestra aplicación en Eclipse

Abriendo el Eclipse

Registrando el proyecto en el tomcat local

Compilando y depurando la aplicación con el Eclipse IDE

Lo que nos queda

Conclusión

Resumen de lo que vamos a ver en el tutorial

- Descargarnos el Maven 2.0 e instalarlo en nuestro PC de desarrollo
- Descargarnos mediante Subversion SVN las actualizaciones de los plugins de Myfaces
- Creación de un proyecto Web con MyFaces y Maven
- Compilación del proyecto mediante Maven y pruebas en un Tomcat 5.5
- Importación del proyecto en Eclipse 3.2 para utilizarlo como entorno de desarrollo y pruebas

Introducción

La creación de aplicaciones Web es una tarea rutinaria en el mundo Java. Aun así, la puesta en marcha de un proyecto de aplciación Web normalmente necesita un trabajo previo de configuración de nuestro entorno, que sin la ayuda de herramientas como los entornos de desarrollo y los asistentes de creación es larga y tediosa. Si además usamos algún framework de desarrollo como por ejemplo JSF, la tarea se complica más… o no.

Para gestionar el ciclo de desarrollo de nuestras aplicaciones java cada vez es más frecuente la utilización de Maven 2.0, ya que esta herramienta se encarga de todos los pasos del desarrollo de una forma sencilla.

Pero además esta herramienta nos puede ayudar en el proceso de creación de nuestro proyecto Java. ¿Cómo lo logra? Mediante el uso de los arquetipos de Maven, que son estructuras de proyecto predefinidas que se generan mediante los plugins de Maven. Además los principales proyectos open source se están volcando en el uso de Maven para su desarrollo, por lo que cada vez es más frecuente encontrar plugins de arquetipos para el desarrollo de aplicaciones.

Nuestro reto va a ser crear una aplicación Web, que utilice el framework JSF en el entorno de desarrollo Eclipse. Pero es que además lo queremos hacer de un modo sencillo y rápido.

En este tutorial vamos a aprender a construir una aplicación básica JSF (Java Server Pages) utilizando el Maven 2.0 y las bibliotecas de MyFaces. Lo mejor de todo es que para crear el ejemplo no vamos a programar ni una línea.

Requisitos

Para este tutorial he utilizado Windows XP, maven 2.0 y Eclipse 3.2. Si no es tu caso, probablemente puedas adaptar fácilimente el tutorial a lo que dispongas.

- Maven 2.0, por ejemplo la distribución maven-2.0.6-bin.zip
- Cliente de subversion, por ejemplo svn-win32-1.4.3.zip
- Eclipse 3.2 correctamente instalado y con las herramientas de desarrollo web WST.
- Tomcat 5.5, y configurado en el Ecplise para poder ejecutar aplicaciones en él..
- Java JDK 1.5 installado, y con la variable JAVA_HOME configurada correctamente

Los ficheros de instalación se pueden descargar de sus respectivas webs:

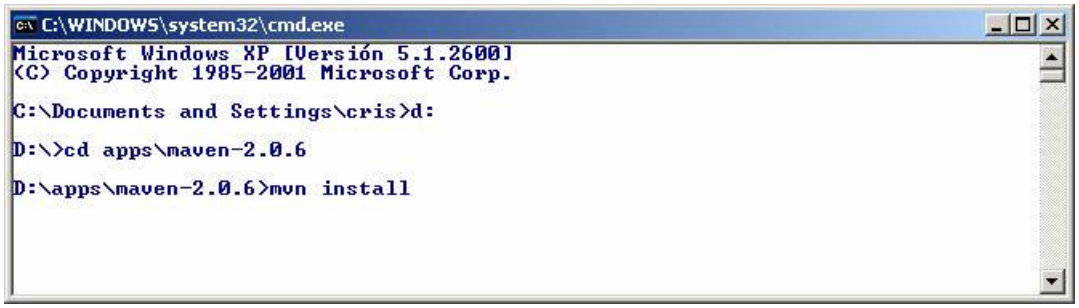
- Maven.apache.org
- Tomcat.apache.org
- www.eclipse.org
- subversion.tigris.org
- java.sun.com

Instalación de Maven

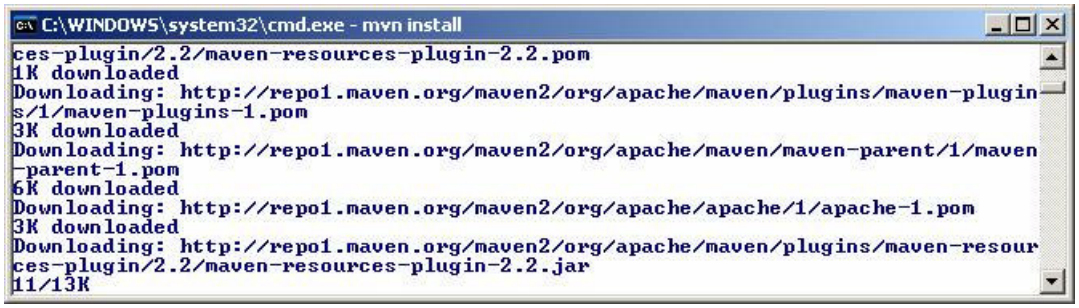
Descoomprimimos el maven a una carpeta (por ejemplo d:\apps\maven2.0.6. Añadimos esta ruta al path del sistema (en Windows hay que añadirlo en propiedades de “mi pc”, en la pestaña opciones avanzadas, botón “Variables de entorno”). Por ejemplo:

Si el JAVA_HOME no está correctamente configurado, hacerlo de igual modo.Recordad que no debe contener espacios en blanco la ruta.

Abrimos una consola (cmd.exe en Windows) e instalamos el maven

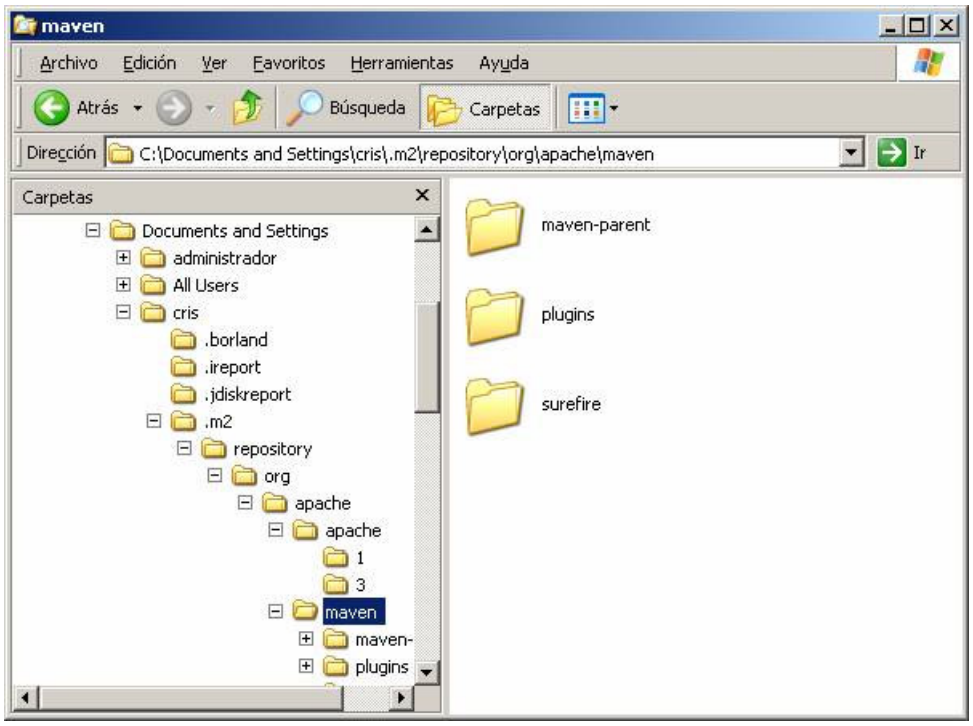


Maven comenzará a instalarse, por lo que veremos bastante actividad en la consola



Al final nos muestra un mensaje de instalación correcta, aunque nos avisa de que hay un error. No se

Fundamentalmente lo que hemos hecho es inicializar el repositorio de maven. En efecto, en nuestra carpeta HOME (en Windows la raiz del perfil del usuario) se crea una carpeta .m2 como la siguiente

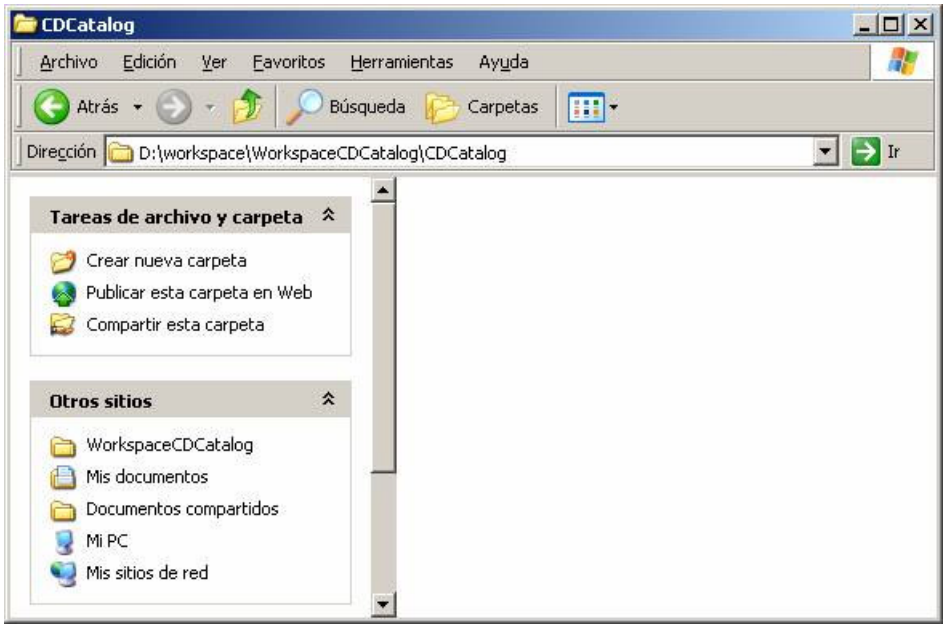


Esta carpeta contiene nuestro repositorio local de maven.

Creación de un proyecto maven 2 para MyFaces

Ahora vamos a crear un proyecto para poder usarlo con el Eclipse 3.2

Crearmos una carpeta en nuestro workspace

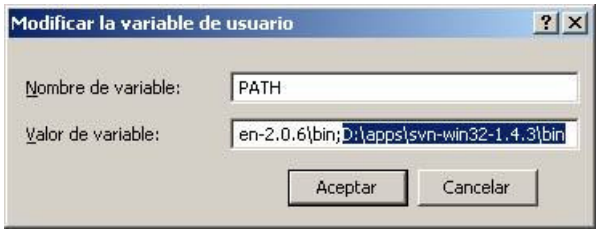


Ahora hay que crear el proyecto. Vamos a dejar que maven lo haga por nosotros. Para ellos MyFaces dispone de un arquetipo maven que construye el proyecto. Sólo necesitamos darle algunos parámetros para que lo cree.

Instalación del arquetipo de myfaces

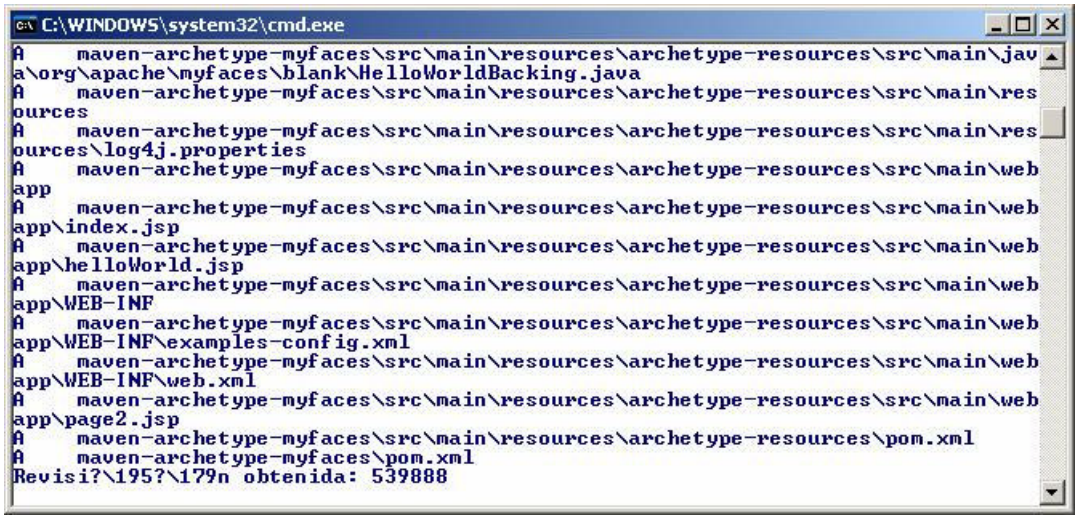
Primero debemos tener un suvbersion instalado. Lo podemos descargar de http://subversion.tigris.org/

La instalación es sencilla. Para windows, descomprimimos el paquete y lo añadimos al path, como siempre

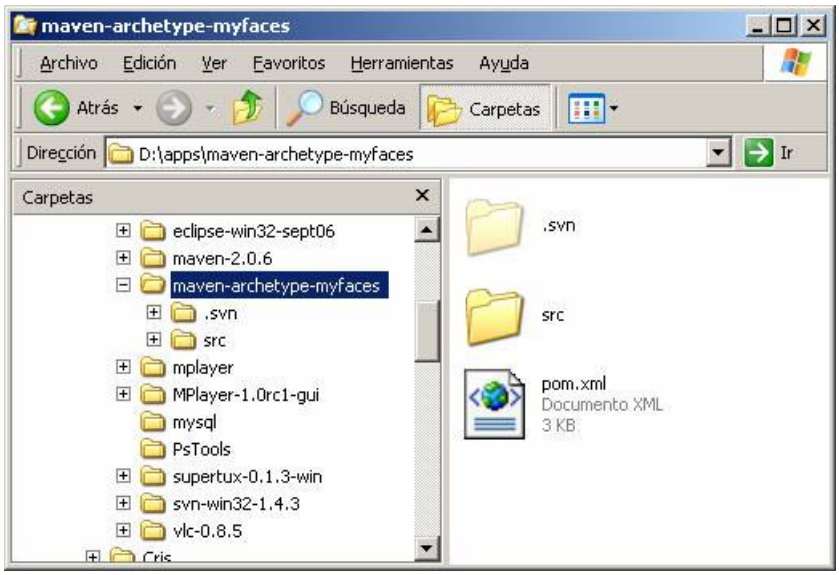


Ahora nos situamos en la carpeta d:\apps para descargarnos el plugin del arquetipo de myfaces.

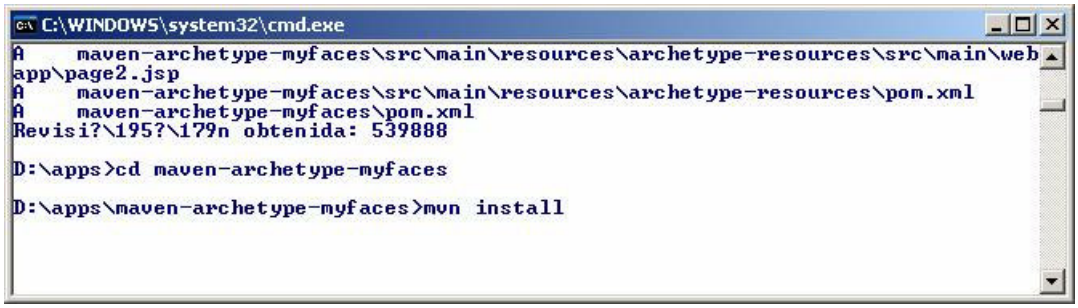
Hasta que termine de descargarlo



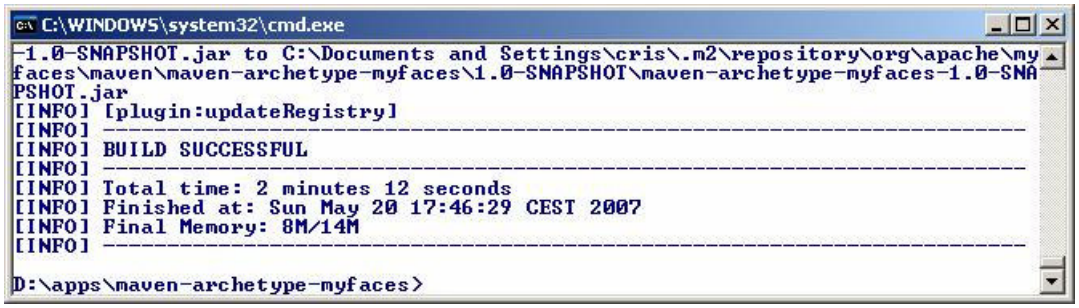
Lo que nos crea la típica carpeta de instalación de proyectos maven.



Nos situamos en ella y lo instalamos



Hasta que se instale



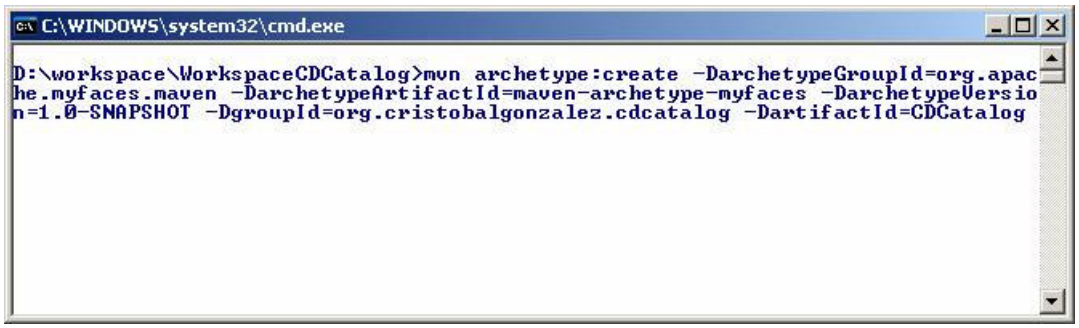
Si revisamos el repositorio maven veremos que se han instalado varios módulos en él.

Creando el proyecto con el arquetipo de maven

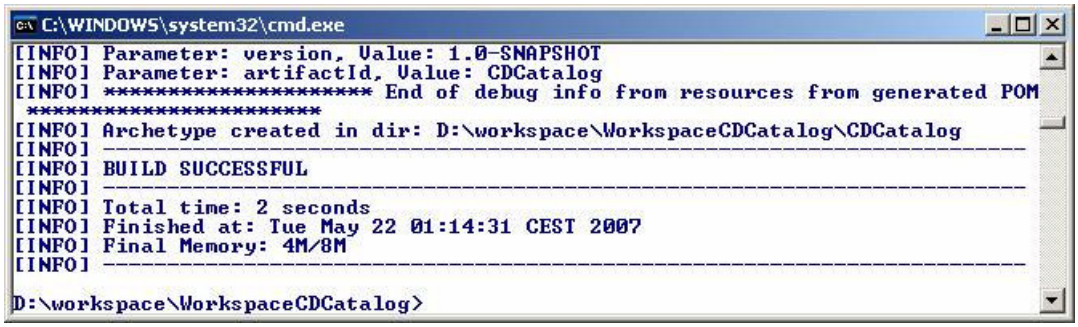
Nos situamos en la carpeta del workspace. Ahora vamos a usar el siguiente comando, que voy a escribir como una sóla línea

```
mvn archetype:create -DarchetypeGroupId=org.apache.myfaces.maven      \
                -DarchetypeArtifactId=maven-archetype-myfaces    \
                -DarchetypeVersion=1.0-SNAPSHOT                \
                -DgroupId=org.cristobalgonzalez.cdcatalog                        \
                -DartifactId=CDCatalog
```

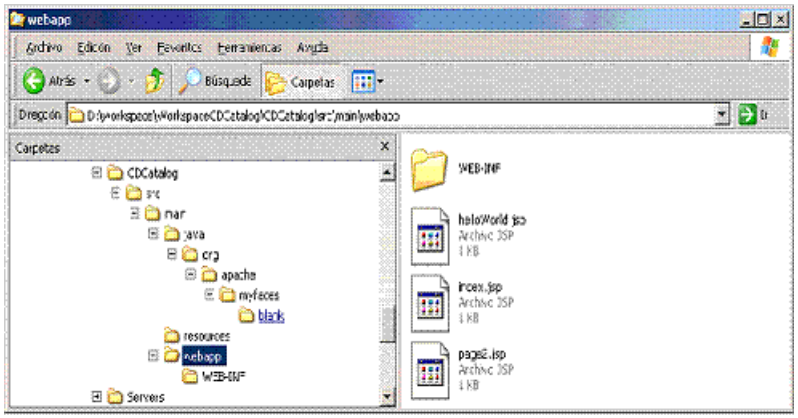
Recordemos que debemos escribirla en la consola como una única linea, muy larga, por cierto. Lo más importante es el groupId y el artifactId, que designan el grupo de aplicaciones a la que pertenece nuestra aplicación y el identificador de la aplicación. El grupo lo especificaremos siguiendo una nomenclatura similar a la usada en los paquetes java.



Pulsamos enter, y maven se pone a trabajar



ahora si miramos la estructura de carpetas que ha creado en el espacio de trabajo veremos las carpetas del proyecto

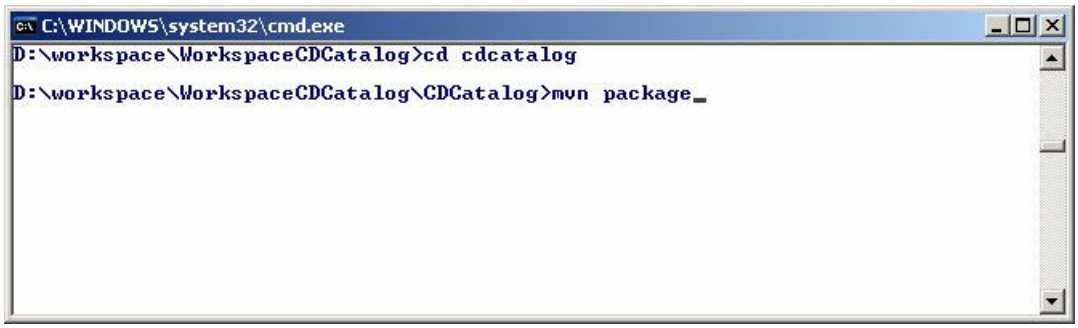


Como vemos se ha creado la estructura básica. Nos ha creado:

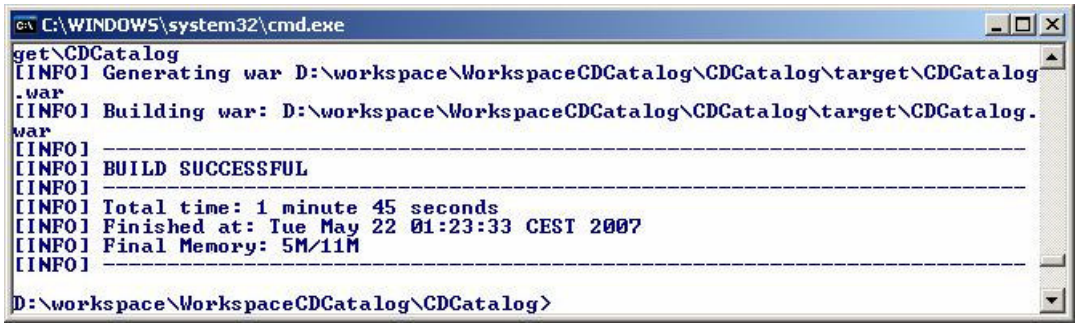
- Jsp de ejemplo para myfaces
- Una clase helloworld.java
- Un web.xml con las entradas de MyFaces asociadas

Compilando la aplicación

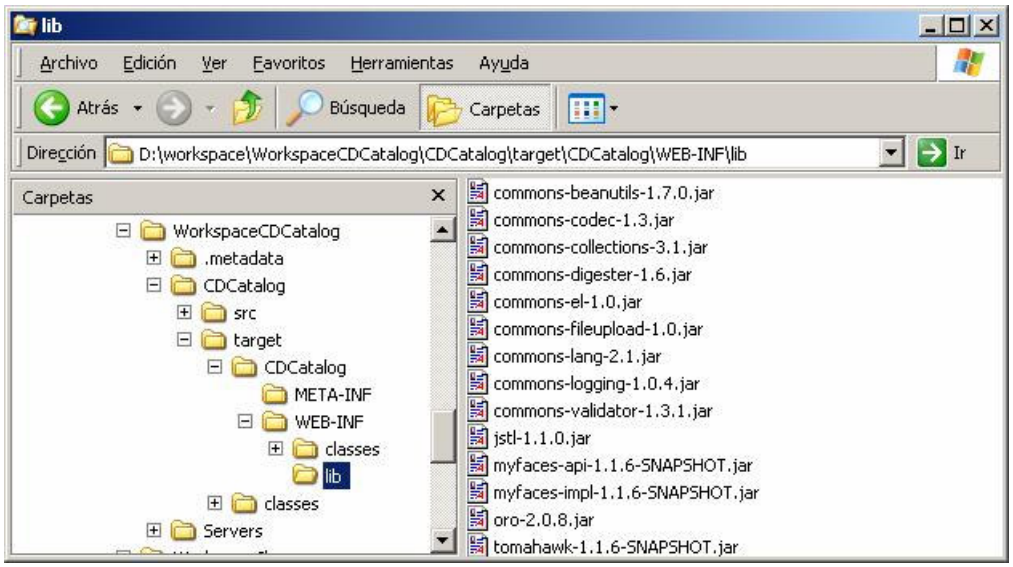
Usamos el maven para compilar la aplicacióin y generar el war



comienza a trabajar maven compilando el proyecto. Si todo ha ido bien, y después de descargarse bastantes cositas, tendremos el mensaje:



Veamos qué nos ha dejado. En la carpeta del proyecto se ha generado la carpeta target, con una carpeta CDCatalog con el contenido del war y un CDCatalog.war, que es el fichero de despliegue de la aplicación

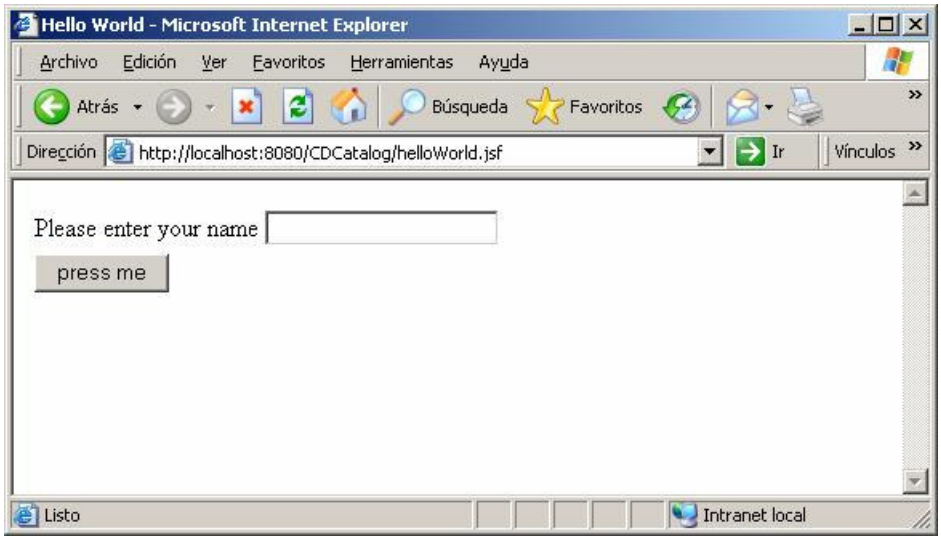


Además, todas las bibliotecas (jars) que necesitamos para la aplicación también se han descargado (algunas hasta se han compilado en el repositorio local)

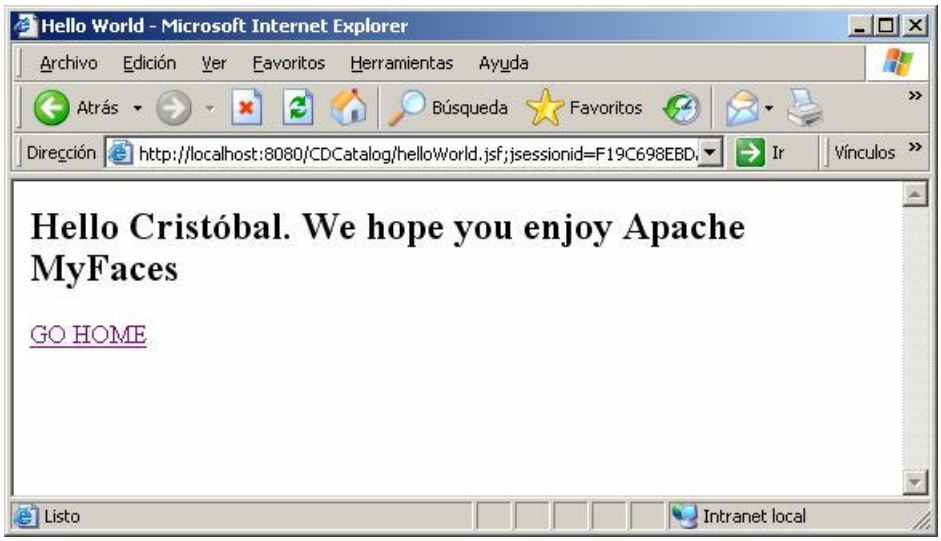
Probando la aplicación

Para probar la aplicación debemos utilizar un servidor de aplicaciones adecuado, por ejemplo Tomcat 5.5. Una vez compilado y generado el war, basta desplegarlo en el servidor web para poder probar la aplicación.

Arrancamos el Tomcat, por ejemplo desde la línea de comandos con startup.bat y luego abrimos la página inicial del proyecto que nos ha creado el plugin



Como siempre por defecto será http://localhost:8080/MiNombreProyecto/index.jsp. Como vemos la página está lista para funcionar como aplicación JSF. Probémosla. Escribimos nuestro nombre y pulsamos el botón.

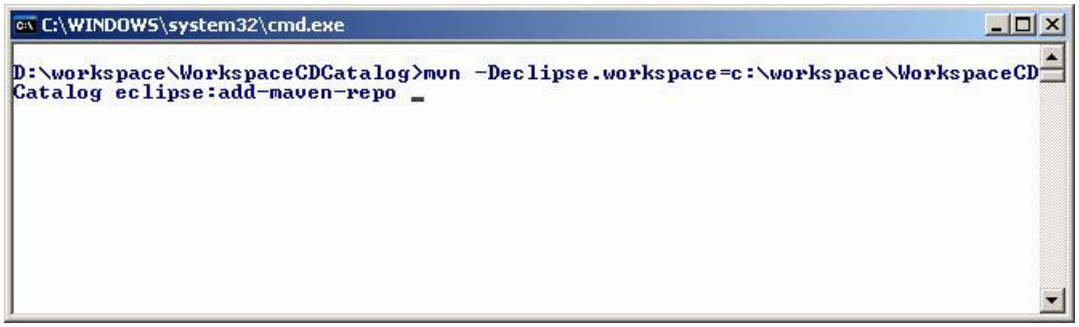


Como vemos ha navegado a otra página. Por defecto la extensión del servlet de JSF de MyFaces es jsf. Habitualmente se suele usar la extensión .faces, pero como siempre esto da lo mismo.

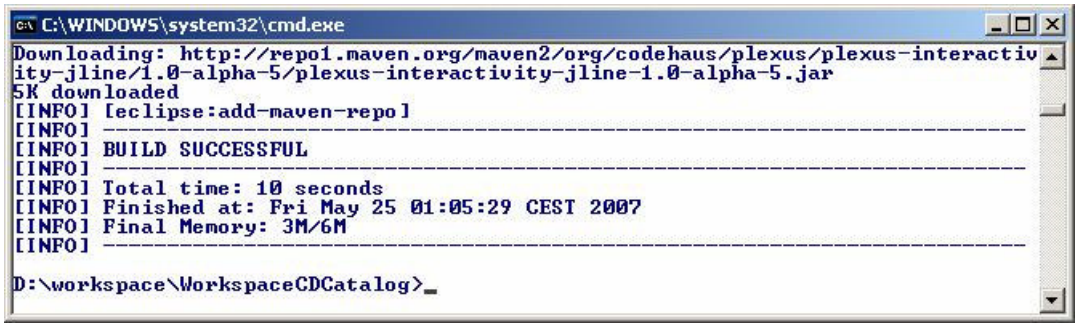
Integrando nuestra aplicación en Eclipse

Lo primero que vamos a hacer es enseñar al Eclipse dónde está el repositorio. Eclipse guarda esta información por workspace, en la carpeta .settings, por lo que el plugin de maven para eclipse dispone de una tarea específica para registrarse en Eclipse. Ejecutamos lo siguiente:

```
D:\ >mvn -Declipse.workspace=RutaAlWorkspace eclipse:add-maven-repo
```

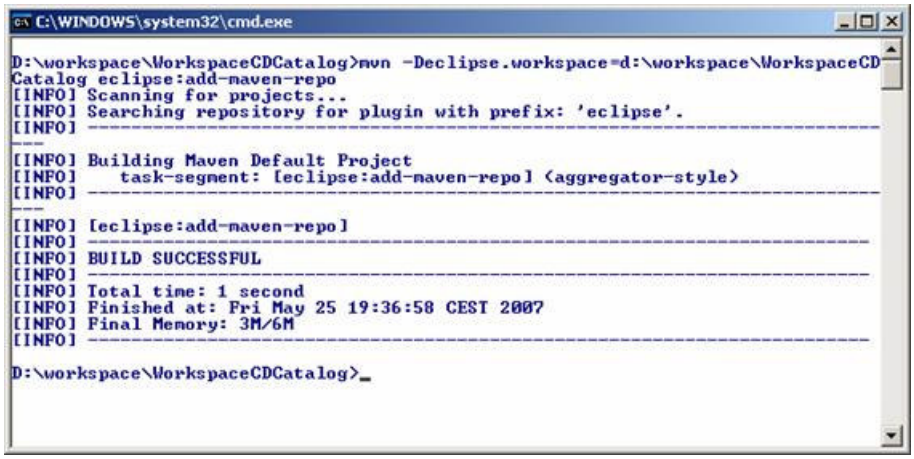


Y vemos lo que ocurre

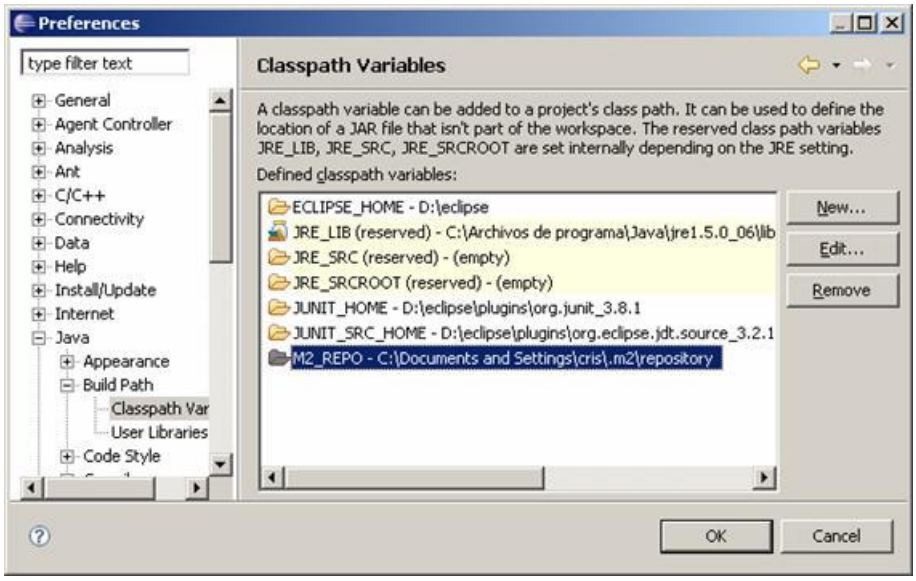


Ahora Eclipse ya sabe donde está el repositorio. Le hemos añadido una variable que apunta a nuestro repositorio local.

En mi caso, después de añadir el proyecto al Eclipse tuve que volver a ejecutar esto, ya que no me había creado la variable M2_REPO en el Eclipse. El síntoma es que Eclipse no podía compilar el proyecto (salían muchos mensajes de error de compilación). Ejecuté de nuevo la tarea de maven para crear la variable.



Y luego lo comprobamos abriendo en Eclipse "Windows\Preferences" en la pestaña Java\Build Path\Classpath variables

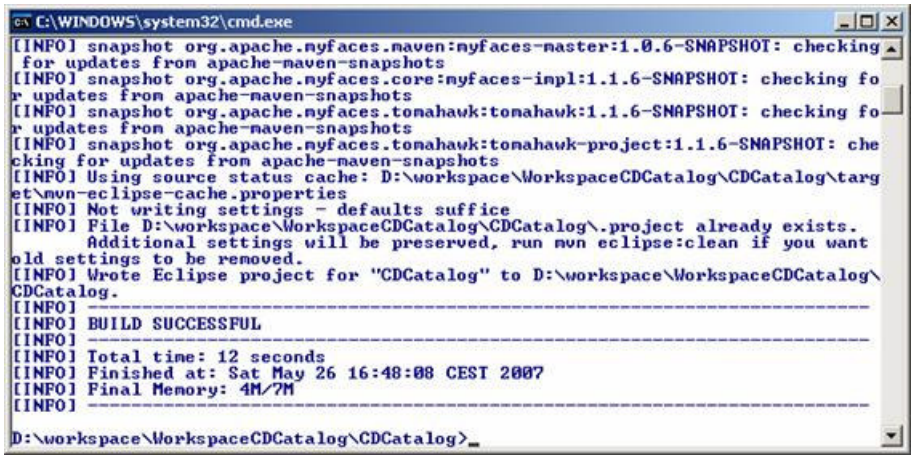


Como vemos se ha creado la variable M2_REPO y el proyecto se compila sin problemas.

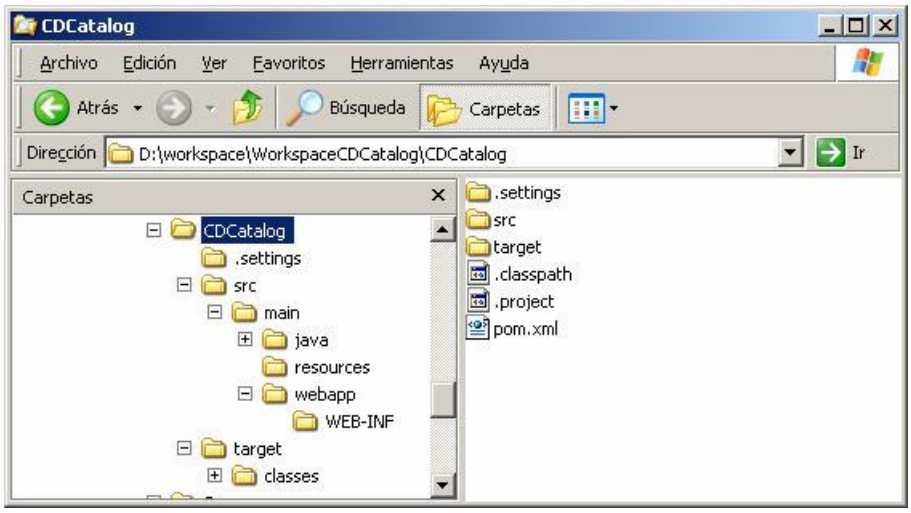
Lo siguiente que hacermos es preparar el proyecto para ser importado en Eclipse. Nos situamos en el proyecto y ejecutamos mvn eclipse:eclipse

Nota: aunque en la documentación del maven no lo dice, yo he usado como parámetro la versión 1.5 del plugin wst. Esto me genera más ficheros de configuración que con la 1.0.

Comienza a trabajar y nos sale el siguiente resultado



En principio dice que ha ido bien, pero que si queremos descargar los códigos fuentes debejos ejecutar la misma tarea con el parámetro –DdownloadSources=true. Lo dejamos para más adelante.



Como vemos en el proyecto nos ha creado el .classpath de Eclipse y el .proyect. Por ejemplo el classpath quedaría así:

.classpath

```
<classpath>

  <classpathentry kind="src" path="src/main/java"/>

  <classpathentry kind="src" path="src/main/resources" excluding="**/*.java"/>

  <classpathentry kind="output" path="target/classes"/>

  <classpathentry kind="con" path="org.eclipse.jdt.launching.JRE_CONTAINER"/>

  <classpathentry kind="var" path="M2_REPO/oro/oro/2.0.8/oro-2.0.8.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-collections/commons-
collections/3.1/commons-collections-3.1.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-el/commons-el/1.0/commons-el-1.0.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-logging/commons-logging/1.0.4/commons-
logging-1.0.4.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-validator/commons-
validator/1.3.1/commons-validator-1.3.1.jar"/>

  <classpathentry kind="var" path="M2_REPO/javax/servlet/jstl/1.1.0/jstl-1.1.0.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-digester/commons-digester/1.6/commons-
digester-1.6.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-codec/commons-codec/1.3/commons-codec-
1.3.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-fileupload/commons-
fileupload/1.0/commons-fileupload-1.0.jar"/>

  <classpathentry kind="var" path="M2_REPO/org/apache/myfaces/core/myfaces-api/1.1.6-
SNAPSHOT/myfaces-api-1.1.6-SNAPSHOT.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-lang/commons-lang/2.1/commons-lang-
2.1.jar"/>

  <classpathentry kind="var" path="M2_REPO/commons-beanutils/commons-
beanutils/1.7.0/commons-beanutils-1.7.0.jar"/>

  <classpathentry kind="var" path="M2_REPO/junit/junit/4.0/junit-4.0.jar"/>

  <classpathentry kind="var" path="M2_REPO/org/apache/myfaces/tomahawk/tomahawk/1.1.6-
SNAPSHOT/tomahawk-1.1.6-SNAPSHOT.jar"/>

  <classpathentry kind="var" path="M2_REPO/org/apache/myfaces/core/myfaces-impl/1.1.6-
SNAPSHOT/myfaces-impl-1.1.6-SNAPSHOT.jar"/>
```

```
</classpath>
```

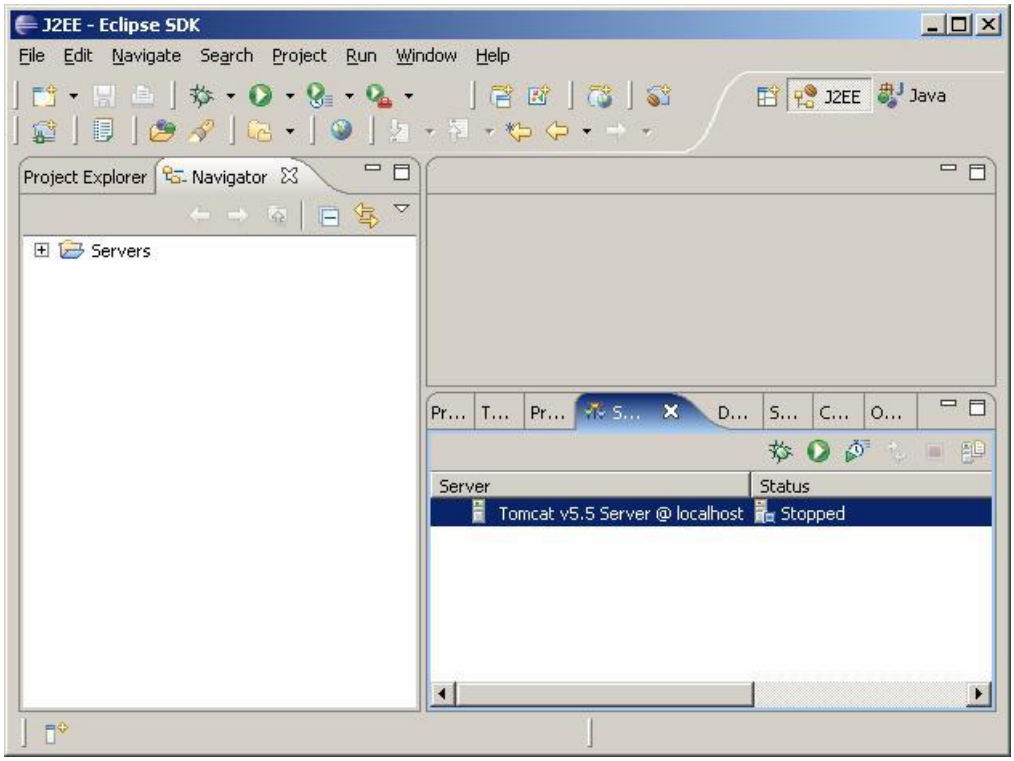
También se nos han creado otros ficheros y carpetas importantes:

- .settings, en la que se ha creado la configuración para el plugin WST del Eclipse
- .project, con la configuración de los natures de Eclipse para el proyecto

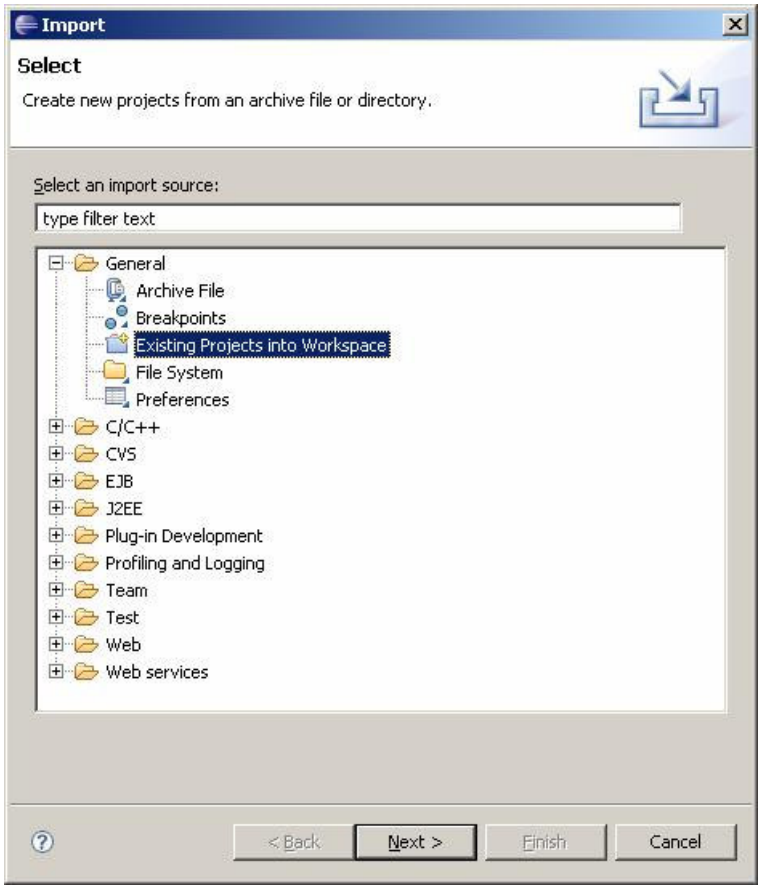
.

Abriendo el Eclipse

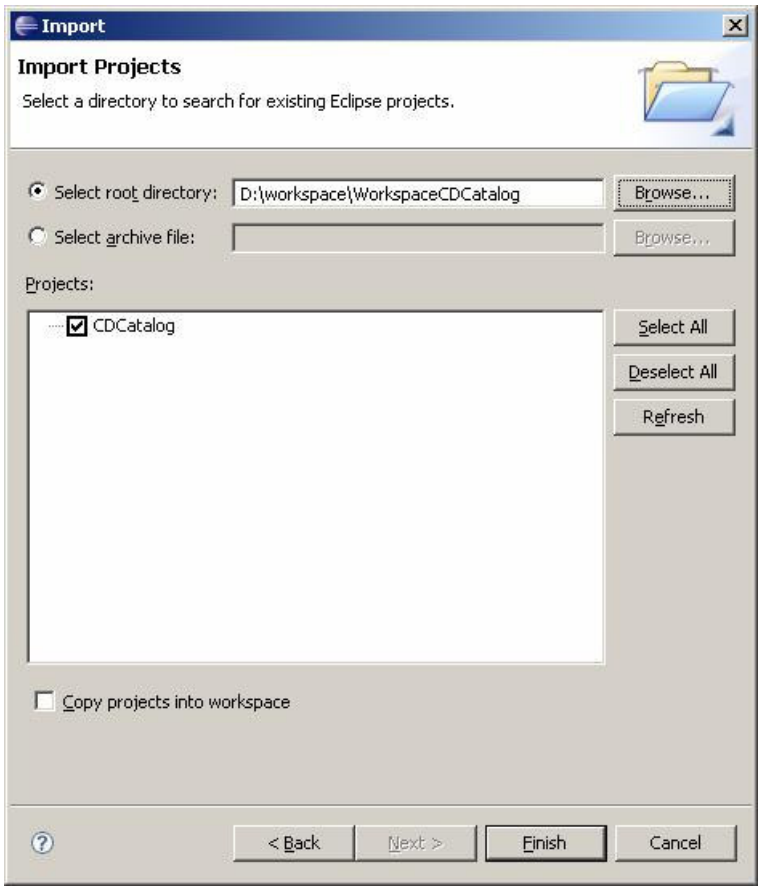
Abrimos ahora el Eclipse



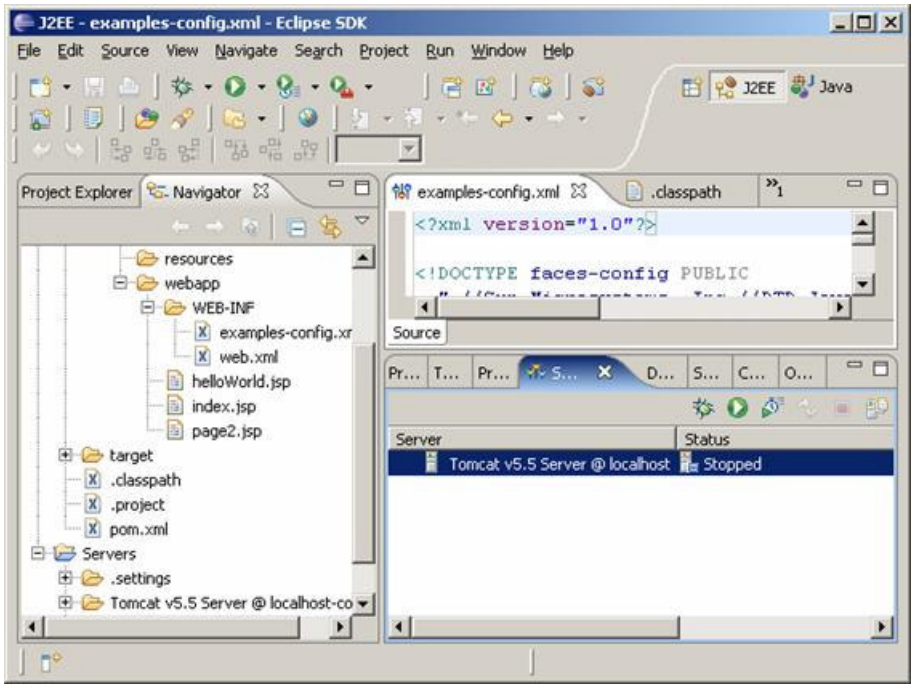
Para importarlo abrimos el menú "File\Import …"



Elegimos la opción "Existing project into workspace"



Navegamos hasta el workspace en el que hemos creado el proyecto, lo seleccionamos y pulsamos "Finish"

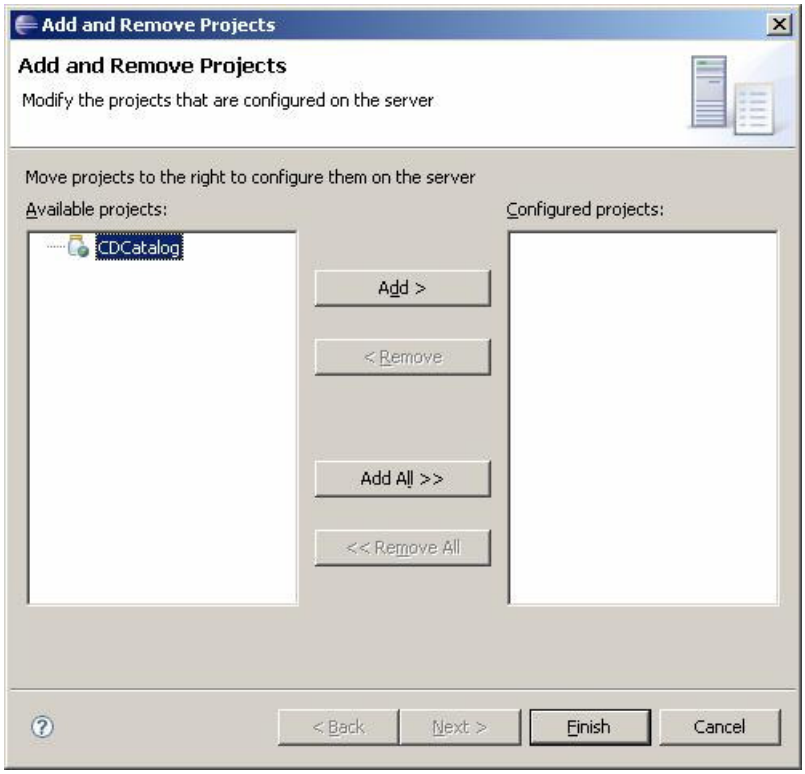


Ya tenemos el proyecto en Eclipse. Desde aquí tendremos varios modos de trabajo:

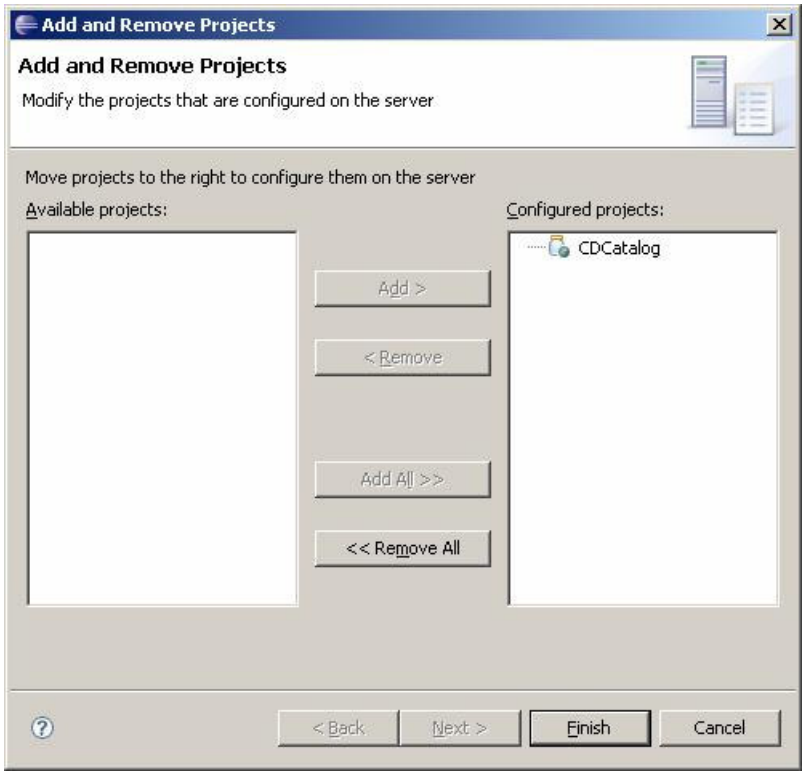
- Usar el modo manual completo. En este caso compilamos y generamos el war con maven, como hemos hecho antes y probamos en un servidor web externo.
- Instalamos el plugin de integración de Eclipse y trabajamos con él.

Registrando el proyecto en el tomcat local

Pulsamos con el botón secundario en el icono del Tomcat en la pestaña "Servers" y elegimos la opción "Add and remover modules…"



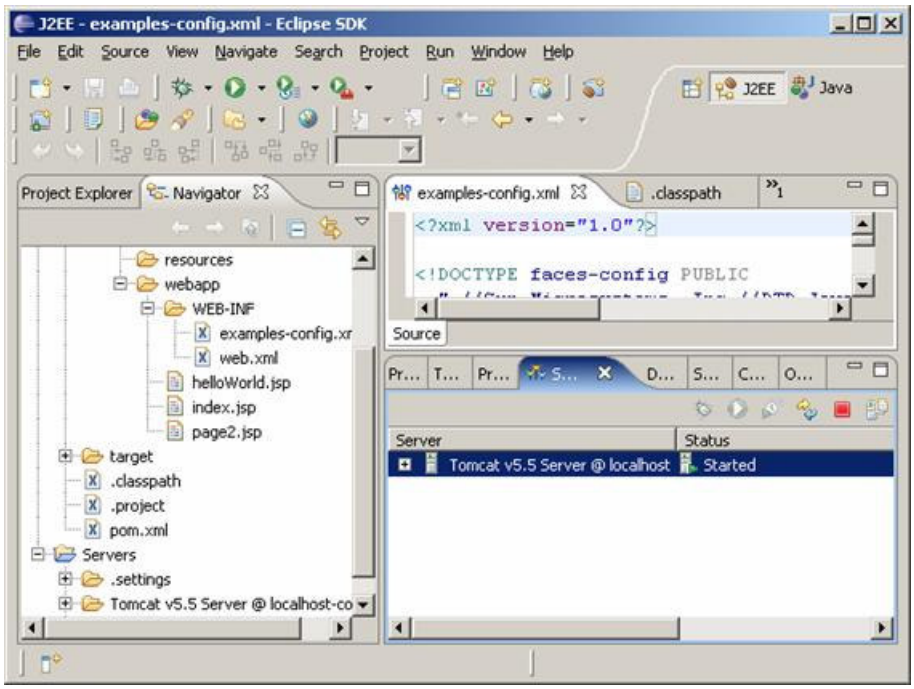
Elegimos nuestro proyecto y lo añadimos



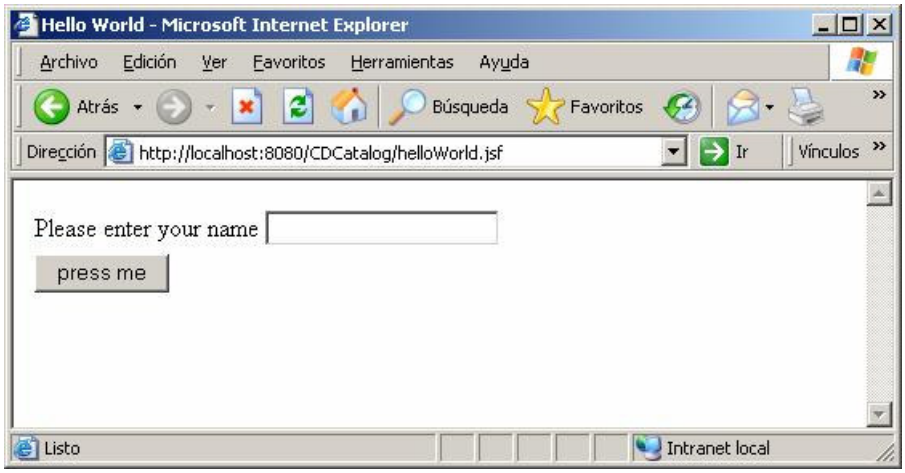
Pulsamos "Finish" y ya queda registrado el proyecto.

Nota, si no te aparece tienes un problema con la configuración del plugin wst, por ejemplo si has usado una versióin 1.0 en vez de la 1.5 del wstversion al crear el proyecto.

Para probarlo arrancamos el tomcat, seleccionando en el icono del Tomcat la opción "Start" con el botón secundario del ratón.



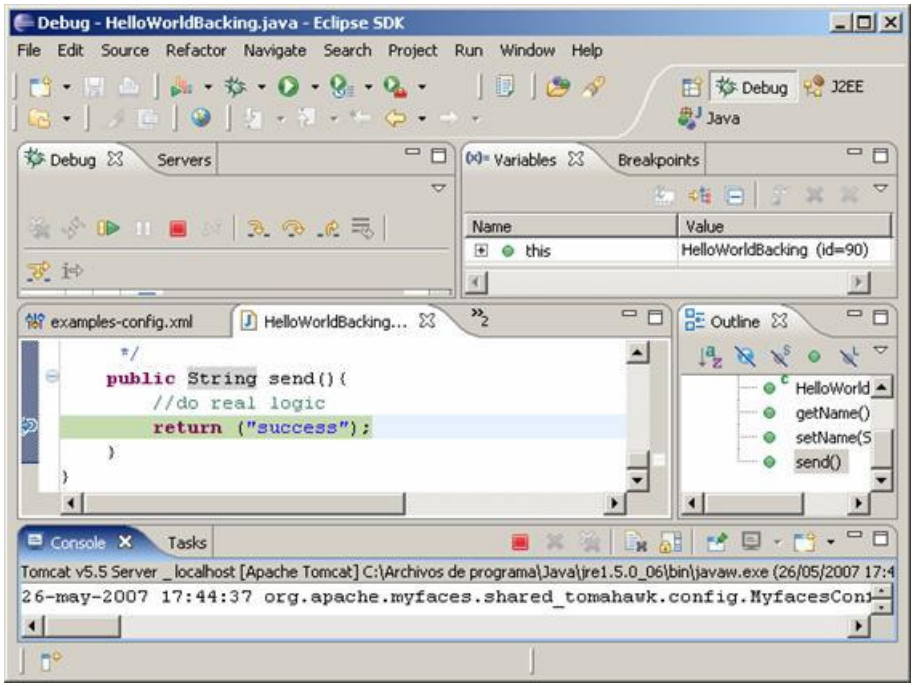
Probamos nuestra aplicación para ver que todo funciona, abriendo un navegador:



Ya la tenemos funcionando.

Compilando y depurando la aplicación con el Eclipse IDE

Para probar el depurador ponemos un punto de ruptura en la clase HelloWorld.java y probamos a ejecutar laplicación web.



Si todo ha ido bien, al ejecutar la aplicación Eclipse cambiará a la perspectiva de depuración y podremos depurar la aplicación.

Lo que nos queda

Nos quedan algunos detalles. Por ejemplo no está correctamente registrado el plugin de JSF del WST del Eclipse. Estop requiere un poco de trabajo porque hay que especificar las bibliotecas de JSF para el plugin en el momento de crear el proyecto, pero eso lo hace nuestro plugin de Maven, que no está preparado para el JSF de Eclipse.

Conclusión

Como vemos utilizzando la herramienta Maven y el framework de desarrollo JSF de MyFaces es fácil crear el esqueleto de una aplicación Web en minutos, utilizando además nuestro entorno de desarrollo favorito para ello. La tarea más tediosa que es crear el esqueleto del proyecto se convierte en una tarea sencilla, que además nos abre un mundo de posibilidades. Añadir bibliotecas al proyecto es muy fácil, ya que al estar la construcción gobernada por Maven, basta añadir las dependencias al proyecto para que Maven se encargue de descargarnos los jar correspondientes.

Si no tienes experiencia en el uso de Maven, con este tutorial podrás comprobar porqué este sistema de desarrollo se está convirtiendo en el favorito de los desarrolladores.

SOME RIGHTS RESERVED This work is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 2.5 License.
Puedes opinar sobre este tutorial aquí

Recuerda

que el personal de Autentia te regala la mayoría del conocimiento aquí compartido (Ver todos los tutoriales)

¿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?

info@autentia.com

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos .....
Autentia = Soporte a Desarrollo & Formación

soluciones reales para su negocio

Autentia S.L. Somos expertos en:
J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ..
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
Mejora de la calidad del código fuente con Eclipse	En este pequeño tutorial se muestra una de esas multiples opciones de la plataforma de desarrollo Eclipse que muchos de nosotros no vemos y que nos hubieran ahorrado un gran esfuerzo si lo hubieramos conocido en su dia.
Pruebas de integración con Maven	Este tutorial nos muestra un ejemplo para lanzar las pruebas de integración "engañando" a Maven para que no se lanzen en la fase de test teniendo únicamente un módulo para ambas
Análisis de rendimiento (Profiling) de aplicaciones web con eclipse	En este tutorial se va a explicar como analizar el rendimiento de nuestras aplicaciones web con una herramienta propia de Eclipse, llamada Eclipse TPTP.
Probando entornos para JSF	En este tutorial os mostramos con ejemplos como utliizar dos conocidos entornos de desarrollo para JSF: Exadel Studio y Sun Studio Creator
Guía de referencia de JSF	Esta guía-tutorial pretende dar a conocer todos los conceptos básicos de JSF así como servir de guía de referencia a los desarrolladores.En ningún caso esta pensada para aprender JSF desde cero.
Crear un repositorio remoto y como hacer una 'release' con varios proyectos en Maven y Eclipse	En este tutorial vamos a explicar como podemos trabajar teniendo varios proyectos relacionados en Maven y en Eclipse
Utilizando JSTL en JSF	Os mostramos como utilizar la librería estandar de etiquetas en JSF, implementando una sencilla aplicación web
Validar en JSF con Commons Validator	En este nuevo tutorial sobre el framework JSF os mostramos como utilizar y extender la validación del Commons Validator
Conversión y validación en JSF	En este nuevo tutorial sobre JSF os mostramos como utilizar y extender los mecanismos básicos de conversión y validación
JSF y comparativa con Struts	Os mostramos los pasos necesarios para empezar a utilizar JSF (Java Server Faces) y su comparación / relación con Struts

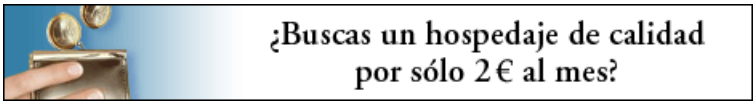
Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

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