

¿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)



[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.



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

Servidores Linux

Optimizar rendimiento y valor de aplicaciones con 32/64 bit
www.Intel.com

BitRock InstallBuilder

Build native Linux, Win32 installer for Java applications. Download now
www.bitrock.com

Software de gestión

Mantenga activas las aplicaciones e infraestructura de la empresa
www.cast-info.es

Anuncios Goooooogle

Como informar a Debian GNU/Linux de nuestra Máquina Virtual Java

Creación: 11-08-2005

Índice de contenidos

- [1. Introducción](#)
- [2. Entorno](#)
- [3. Instalamos los paquetes necesarios](#)
- [4. Creamos los paquetes "dummy" de Java](#)
- [5. Instalación de nuestros paquetes "dummy"](#)
- [6. Probar que todo ha ido bien](#)
- [7. Conclusiones](#)
- [8. Sobre el autor](#)

1. Introducción

Hoy en día es muy normal utilizar alguna aplicación escrita en Java. Para poder ejecutar este tipo de aplicaciones es necesario tener instalada una Máquina Virtual de Java (a partir de ahora la nombraremos como JVM) en nuestro sistema.

En Debian tenemos varias opciones para hacer esto. De hecho podemos encontrar entre los paquetes oficiales varias JVM, como: gij (intérprete de byte code), ikvm (JVM para .NET), jamvm, kaffe, ...

En mi caso he instalado una JVM que no está entre los paquetes de Debian. En concreto he instalado la JVM de Sun Microsystems. Este tipo de JVM no las encontraremos en los paquetes oficiales de Debian por problemas de licencias, ya que no son compatibles con el Software Libre.

Al haber instalado una JVM fuera de los paquetes de Debian, mi sistema desconoce que tiene la capacidad para ejecutar aplicaciones Java.

Lo que vamos a ver en este tutorial es precisamente eso, como indicar a nuestro sistema Debian que tenemos una JVM instalada.

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Ahtex Signal X-9500M (Centrino 1.6 GHz, 1024 MB RAM, 60 GB HD).
- Sistema Operativo: GNU / Linux, Debian Sid (unstable), Kernel 2.6.12, KDE 3.4
- Máquina Virtual Java: JDK 1.5.0_03 de Sun Microsystems

3. Instalamos los paquetes necesarios

Lo primero que vamos a hacer es instalar java-common y equivs

```
# apt-get -u install java-common equivs
```

Con equivs lo que vamos a hacer es crear unos paquetes que sólo sirven para satisfacer dependencias. Es decir, al instalar estos paquetes el sistema sabrá que "alguien" provee cierta funcionalidad. Vamos a denominar a estos paquetes "paquetes dummy" ya que realmente no tienen contenido.

El paquete java-common lo necesitamos porque los paquetes "dummy" que vamos a crear dependerán de este. Además trae documentación muy interesante.

4. Creamos los paquetes "dummy" de Java

Para esto nos movemos a /usr/share/doc/java-common/dummy-packages (aunque la documentación de java-common recomienda trabajar sobre /var/install/java/pkg).

```
# cd /usr/share/doc/java-common/dummy-packages
```

Ahora hacemos:

```
# equivs-build java2-runtime-dummy.control  
# equivs-build java2-compiler-dummy.control  
# equivs-build java-virtual-machine-dummy.control
```

Con esto hemos creado tres paquetes listos para instalar. Una vez instalados nuestro sistema sabrá que tenemos un entorno de ejecución y de compilación basado en Java2, por lo que podremos instalar cualquier paquete que necesite estas dependencias, por ejemplo freemind (estupenda aplicación para representar árboles de pensamiento).

5. Instalación de nuestros paquetes "dummy"

Tan sólo tenemos que hacer:

```
# dpkg -i java2-runtime-dummy_1.0_all.deb  
# dpkg -i java2-compiler-dummy_1.0_all.deb  
# dpkg -i java-virtual-machine-dummy_1.0_all.deb
```

6. Probar que todo ha ido bien

Si no hemos tenido ningún problema, podemos probar si las dependencias están correctamente dadas de alta en nuestro sistema. Para ello vamos a probar a instalar freemind

```
# apt-get -u install freemind
```

Si todo está bien, el paquete se instalará sin ningún problema, y podremos ejecutarlo con:

```
$ freemind
```

Si al ejecutar tenemos algún problema, comprobar que la instalación de vuestra JVM es correcta. Es decir, como mínimo debería estar definida la variable de entorno JAVA_HOME, y los ejecutables de la JVM deberían estar en el PATH.

7. Conclusiones

En este pequeño tutorial hemos visto como informar a nuestro sistema sobre ciertas "habilidades" que están resultas por componentes fuera de la estructura habitual de paquetes. Aunque lo hemos visto para el ejemplo concreto de Java, lo podríais hacer para cualquier dependencia.

Creo que es aconsejable que siempre que instalemos una JVM fuera de los paquetes de Debian, sigamos los pasos de este tutorial para que el sistema tenga constancia de ello.

Por último os recomiendo leer la documentación que viene con el paquete java-common. No es muy extensa y nos explica cual es la

política de Debian frente a Java (además de muchas otras cosas ;).

8. 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, consultoría o desarrollo de piezas a medida puede contactar con



[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	<input type="text"/>
	<input type="button" value="Enviar"/>

Otros Tutoriales Recomendados ([También ver todos](#))

Nombre Corto	Descripción
Fuentes Gnome (GTK) con KDE (Qt)	Alejandro Pérez nos enseña como configurar las fuentes de aplicaciones Gnome (GTK) si usamos KDE (Qt)
Gestión de Errores con Bugzilla	Alejandro Pérez nos enseña como instalar (en Debian) y utilizar Bugzilla, una herramienta gratuita de gestión de errores.
Optimización Java con Eclipse Profiler Plugin	Alejandro Pérez nos enseña como analizar el rendimiento de nuestras aplicaciones con Eclipse Profiler Plugin.
CRM: GESTIÓN DE LAS RELACIONES CON CLIENTES	Cristhian Herrera comparte con nosotros su trabajo de fin de estudios sobre CRM. Es un extenso trabajo que seguro os gustará
Introducción al UML	Este es el primer artículo sobre el diseño de proyectos orientados a objeto con UML, donde se describe los primeros diagramas a utilizar
Firma digital de un Applet	Para que un applet Java pueda ejecutarse en un cliente Web con la configuración de seguridad por defecto y/o adquirir privilegios de seguridad, es necesario firmarla digitalmente. Alejandro Perez nos enseña como hacerlo de un modo rápido y sencillo.
Manejar dos bases de datos distintas con Hibernate	Alejandro Pérez nos enseña como manejar dos bases de datos distintas con Hibernate
Instalación LAMP en Debian	Alejandro Perez nos enseña como instalar nuestro entorno Apache, MySQL y php en Debian
Auto-gestión de dispositivos USB en Debian	En este tutorial y gracias a HAL (Hardware Abstraction Layer) vamos a ver como se puede gestionar de forma automática cualquier dispositivo que tengamos conectado en nuestro ordenador, discos duros, USB, firewire, ...
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

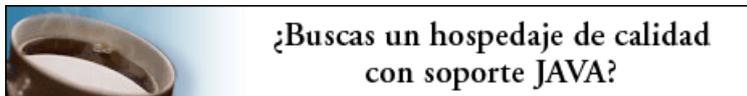
Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

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 Optimizado 800X600