Instalando Java e Eclipse em Linux

Um guia prático para preparar um ambiente de programação Java corretamente em Linux

Este documento (mais completo) na Internet: http://avi.alkalay.net/linux/docs/java/

Veja também http://ibm.com/developerWorks

Veja também http://OpenPowerProject.com/br

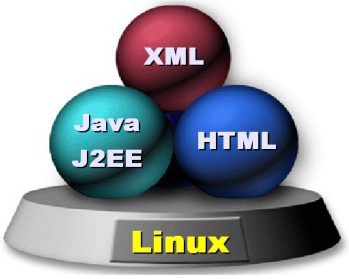
Avi Alkalay

avix@br.ibm.com

Consultor de Linux e Padrões Abertos

Índice

1. Porque Java Com Linux ?



Nos primórdios das tecnologias, todas elas nasciam proprietárias porque seus criadores queriam explora-las ao máximo, por serem todas novidades.

Depois da popularização do PC, e mais ainda, da Internet, fabricantes começaram a se reunir ao redor de Padrões Abertos para criar uma rede de valor onde todos — fabricantes e usuários — acabam ganhando.

Existem hoje inúmeros Padrões Abertos, mas os que se destacam são os seguintes:

- **HTML**
  É a representação universal de interfaces com usuários. Hoje qualquer usuário de computador sabe usar um browser e navegar através de um hipertexto. HTML, ou melhor ainda, hoje, DHTML ou AJAX, é o padrão aberto para aplicações interagirem com usuários.
- **XML**
  Antes de XML, não havia um padrão aberto amplamente aceito que permitisse qualquer aplicação falar com qualquer outra aplicação, mesmo de fabricantes diferentes. XML se tornou a base dos Web Services e Arquitetura Orientada a Serviços, que traz o benefício da integração de processos, com parceiros, clientes e fornecedores.
- **Java e JEE**
  Java é a tecnologia escolhida por toda a indústria para transformar processos de negócio em software. É o Padrão Aberto para se escrever aplicações. Antes de Java, desenvolvedores usam diversas linguagens, sem uma metodologia universal de programação e sem nenhum padrão de bibliotecas de alto nível. JEE (Java Enterprise Edition) é um padrão de biblioteca com métodos universais para aplicações de negócio.
- **Linux**

É o sistema operacional escalável e multiplataforma para rodar tudo isso. É o componente aberto que faltava para ligar a lógica de negócio com padrões abertos de HW.

Essas quatro tecnologias juntas provém tudo que um desenvolvedor precisa para criar suas aplicações de negócio.

2. Java Comparado A C/C++, PHP, Perl E Python

Cabe ao desenvolvedor escolher a linguagem/tecnologia certa para a aplicação certa. Não só os aspectos tecnológicos devem ser levados em conta, mas também aceitação no mercado, aderência a padrões, reputação, política de atualização da tecnológica, prontidão para uma aplicação de negócios, etc.

- C é uma linguagem criada para desenvolver sistemas operacionais, ou algoritmos de baixo nível, quase no nível da máquina, e é nesse nível que essa linguagem se sai melhor. C++ surgiu a alguns anos trazendo orientação a objetos, mas ambas linguagens falharam em padronizar suas semânticas e, principalmente, bibliotecas multiplataforma abertas, e de uso genérico. A não ser que você esteja escrevendo sistemas operacionais, ou bibliotecas de acesso a hardware, uma linguagem mais prática que C ou C++ deve ser escolhida para desenvolver sua aplicação de negócio.
- PHP é uma linguagem/tecnologia desenhada para criar páginas web dinâmicas. Seus programas são geralmente mesclados com código HTML e equivale a JSP e ASP. É muito usada e provou seu valor, porém tem pouca penetração no mundo corporativo e de aplicações de negócio (de fabricantes de SW), e por isso pouco suporte da indústria para que a tecnologia evolua como um padrão. Então, por ser um investimento de risco, dificilmente uma grande empresa vai escolher PHP como tecnologia estratégica para a confecção de suas aplicações críticas, mesmo porque PHP é mais madura somente para aplicações web.
- Perl é abreviação de Practical Extract and Reporting Language, que sugere ter sido criada para manipular texto. A linguagem e suas bibliotecas cresceram para muito além disso, e há hoje quem a use para fazer grandes sistemas. Porém isso é considerado um exagero de uso, pois os programas são interpretados em tempo de execução, o que acarreta performance limitada, e é de fato desenhada para automatizar tarefas de sistema operacional. Python, apesar de ser mais moderna e poder ser compilada, não foge muito deste escopo também. Além disso, ambas não conseguiram uma aceitação comercial madura, e, não representando um investimento seguro a longo prazo, não devem ser escolhidas como estratégicas para a fábrica de SW de uma empresa, ou para um sistema complexo e de missão crítica.

Em contrapartida, a tecnologia Java tem as seguintes características:

- Atingiu um nível de maturidade e aceitação de toda a industrial que o torna um investimento seguro quando da escolha de uma plataforma de desenvolvimento de aplicações de negócio.
- Evolui de acordo com as decisões de um comitê independente chamado Java Community Process, onde empresas e indivíduos votam igualmente para a aceitação de uma novidade. São integrantes ativos do JCP empresas como IBM, Apache Software Foundation, Dolby Laboratories, JBoss, SAP, Oracle, Nokia, Sony, etc. Lista completa em http://jcp.org/en/participation/members
- Toda a indústria respeita as decisões do JCP, evitando o surgimento de derivados (forks) de comportamento diferente.
- É um grande polo tecnológico, tendo somente .NET como seu polo oposto e concorrente (e ainda imaturo de certa forma).

3. Instalando Java Em Linux

Há muitas formas de instalar a JVM em Linux, mas há somente uma forma correta: usando RPM através do repositório JPackage.

3.1. Sobre Repositórios De RPMs

A instalação de um pacote RPM pode falhar se outro pacote precisa ser instalado antes. Isso é conhecido como o inferno das dependências.

Para resolver este problema a comunidade criou ferramentas de instalação de pacotes como o Yum e o APT, que, junto com os metadados oferecidos por um repositório de RPMs, liquidam este problema calculando tudo que é necessário fazer para instalar certo pacote, atualizando automaticamente pacotes já instalados, ou instalando novos, tudo para satisfazer as dependências do pacote que o usuário deseja instalar.

Um repositório é um site na web que contem vários RPMs e metadados de interdependências sobre esses pacotes, que são usados por ferramentas como yum e apt-get.

3.2. O Projeto JPackage E Seu Repositório De RPMs

O JPackage é um repositório de RPMs de alta qualidade de softwares relacionados a Java. É uma comunidade de pessoas que empacotam em RPM as JVMs mais conhecidas do mercado, bem como softwares Java populares como Tomcat, Eclipse, Jakarta, etc.

A primeira pergunta que surge depois que dizemos isso é: **"Mas as JVMs da Sun, IBM, etc já não são disponibilizadas em RPM ?"** Sim, mas cada fornecedor empacota como bem entende, sem seguir nenhum padrão de diretórios ou do sistema operacional. E essa despadronização faz a tecnologia como um todo ser mais difícil de usar.

O Projeto JPackage resolveu isso definindo uma organização de diretórios que permite multiplas JVMs, e lugares padronizados para arquivos JAR, WAR, EAR, etc. O JPackage inovou simplesmente aplicando os conceitos do Filesystem Hierarchy Standard — um padrão aberto dos mais importantes para Linux — aos softwares Java.

O resultado é tão bom, que a Red Hat, SUSE, Mandriva e outros adotaram o padrão JPackage de empacotamento e diretórios para tudo que se refere a Java em suas distribuições (RHEL, Fedora, SLES, SLED, OpenSUSE, NLD, Mandriva, etc).

3.3. Problemas Do JPackage

O JPackage tem uma diretriz de fornecer em seu repositório somente RPMs de softwares livres. Por isso, softwares que não tem licenças livres estão lá somente como RPMs-fonte, que não são tão simples de se instalar, mas mesmo assim promovem a organização e a qualidade do JPackage. Entre esses softwares estão a própria JVM, que vamos demonstrar sua instalação agora.

3.4. Inicializando O JPackage Em Seu Sistema

Antes de instalar qualquer RPM oferecido pelo JPackage, você precisa configurar as ferramentas que acessam e instalam os pacotes automaticamente no seu sistema.

Nos nossos exemplos, vamos usar o Fedora Linux com YUM. Pode-se optar pelo apt-get ao invés do YUM, ou de outra distribuição Linux ao invés do Fedora. No caso do Red Hat Enterprise Linux ou CentOS, o processo é idêntico.

Tenha O YUM Ou Apt-get No Seu Sistema

No caso do Fedora 4, RHEL 4 ou CentOS 4, já temos o YUM instalado no sistema, e só teremos que configura-lo.

No caso de outro Linux, você pode testar se estas ferramentas estão instaladas simplesmente executando o comando **`yum`** ou **`apt-get`**.

Se você finalmente concluiu que não as tem, encontre-as aqui:

- Download do Yum: http://linux.duke.edu/projects/yum/download.ptml
- Download do apt-get: https://moin.conectiva.com.br/AptRpm

Nos nossos exemplos, vamos usar o Yum.

Configure O YUM Para Usar O Repositório JPackage

Basta instalar um arquivo de configuração no diretório **/etc/yum.repos.d/** desta maneira:

```
bash# cd /etc/yum.repos.d/
bash# wget http://www.jpackage.org/jpackage.repo
```

Edite o arquivo **jpacakge.repo** que você acabou de baixar habilitando e desabilitando os canais de RPMs específicos para seu sistema. Por exemplo, no nosso Fedora Core 4, garantimos que os canais **jpackage-generic** e **jpackage-fc** contém a linha ***"enabled=1"***.

Instale O Primeiro Pacote

O pacote **jpackage-utils** deve estar instalado para começar usar o repositório. Nas últimas versões das distribuições populares, ele já está instalado. Nesse caso é boa idéia atualiza-lo.

Para fazer isso:

```
bash# yum install jpackage-utils   # No caso de não estar instalado ainda.
bash# yum update jpackage-utils    # Para atualiza-lo.
```

3.5. Instalando A Máquina Virtual Java (JVM)

Esta é uma das partes mais difíceis porque por questões de licensa o Projeto JPackage não tem permissão para prover o RPM pronto para ser instalado de softwares que tem licensa restrita. É o caso de todas as JVMs comerciais. O JPackage provê o pacote fonte que a partir dele pode-se construir fácil, porém manualmente, o RPM instalável. E vamos demonstrar isso aqui.

JVM Da IBM

Seguimos estes passos:

1. http://www.jpackage.org
2. Procuramos e baixamos o nosrc.rpm da JVM da IBM. A última vez que olhamos estava em http://mirrors.dotsrc.org/jpackage/1.6/generic/non-free/SRPMS/java-1.5.0-ibm-1.5.0.2.3-3jpp.nosrc.rpm
3. Consultamos o pacote para descobrir de onde se baixa a JVM da IBM com o comando **`rpm`**:

```
bash# rpm -qpi java*nosrc.rpm
Name        : java-1.5.0-ibm                Relocations: (not relocatable)
Version     : 1.5.0.2.3                         Vendor: JPackage Project
Release     : 3jpp                           Build Date: Tue 15 Aug 2006
Install Date: (not installed)                Build Host:
tortoise.toronto.redhat.com
Group       : Development/Interpreters       Source RPM: (none)
Size        : 395165271                         License: IBM Binary Code License
Signature   : (none)
Packager    : Thomas Fitzsimmons <fitzsim@redhat.com>
URL         : http://ibm.com/developerworks/java/jdk/linux/download.html
Summary     : IBM Java Runtime Environment
Description :
This package contains the IBM Java Runtime Environment.
```

e descobrimos que devemos procurar na URL marcada.

4. Fomos para http://ibm.com/developerworks/java/jdk/linux/download.html, nos registramos, escolhemos baixar a SDK 1.5 (que é a versão do RPM) em formato tar-gzip (tgz). Tivemos que baixar também a biblioteca javacomm do mesmo lugar. No fim copiamos tudo para o diretório de fontes para RPMs assim:

```
bash# cd /diretorio/onde/baixei/SDK
bash# cp ibm-java2-sdk-50-linux-i386.tgz /usr/src/redhat/SOURCES
bash# cp ibm-java2-javacomm-50-linux-i386.tgz /usr/src/redhat/SOURCES
```

No SUSE, copie para **/usr/src/rpm/SOURCES**.

5. Construimos os pacotes finais com este simples comando:

```
bash# cd /diretorio/onde/baixei/nosrc.rpm
bash# rpmbuild --rebuild java*nosrc.rpm
```

e vimos uma série de coisas acontecendo: é a construção do pacote.

6. Quando terminou, encontramos todos os pacotes gerados em **/usr/src/redhat/RPMS/i386**. Instalamos todos assim:

```
bash# cd /usr/src/redhat/RPMS/i386
bash# rpm -Uvh java*ibm*rpm
```

e a JVM da IBM está instalada.

O padrão JPackage definiu que a JVM deve ser a soma de uma série de sub-pacotes, todos com nome padronizado, e os que geramos neste exemplo são:

java-1.5.0-ibm-1.5.0.2.3-3jpp.i386.rpm	A JRE mínima. É o pacote básico que você deve instalar.
java-1.5.0-ibm-alsa-1.5.0.2.3-3jpp.i386.rpm	Suporte a arquitetura de audio ALSA do Linux.
java-1.5.0-ibm-plugin-1.5.0.2.3-3jpp.i386.rpm	Java Plugin para os browsers Mozilla e Firefox. Não obrigatório.
java-1.5.0-ibm-devel-1.5.0.2.3-3jpp.i386.rpm	O compilador Java e a SDK. Instale-o se você vai programar em Java.
java-1.5.0-ibm-src-1.5.0.2.3-3jpp.i386.rpm	Fontes de programas em Java, para estudo e teste.
java-1.5.0-ibm-jdbc-1.5.0.2.3-3jpp.i386.rpm	Driver JDBC genérico para o unixODBC genérico. Não é necessário se você vai usar o driver JDBC de seu banco de dados.
java-1.5.0-ibm-demo-1.5.0.2.3-3jpp.i386.rpm	Alguns programas demo. Não é obrigatório.
java-1.5.0-ibm-javacomm-1.5.0.2.3-3jpp.i386.rpm	Java Communications API para Linux.

No JPackage há modelos de empacotamento (src.rpm) das JVMs da IBM, Sun, BEA e Blackdown. Para instalar qualquer uma delas, você terá que construir o RPM como demonstramos aqui.

A diferença entre elas está no nome do RPM (“ibm”, “sun”, “blackdown”), e você pode ter instalado em seu sistema JVMs de vários fornecedores simultaneamente. Os RPMs de todos os fornecedores, segundo o padrão JPackage, obedecem esta mesma convenção de nomes de sub-pacotes.

3.6. Instale Outros Softwares Java Que Não Tem Fonte

Será necessário instalar outros RPMs sem fonte para usar corretamente outros pacotes populares do JPackage. Tentanto instalar o tomcat, verificamos que ele necessita do JTA, que é uma API de transações.

Então repetimos os conceitos do passo anterior:

1. Começamos em http://jpackage.org

2. Procuramos e baixamos o nosrc.rpm da JTA. A última vez que olhamos estava em http://mirrors.dotsrc.org/jpackage/1.6/generic/non-free/SRPMS/jta-1.0.1-0.b.4jpp.nosrc.rpm

3. Consultamos o pacote (ou as infos sobre o pacote em jpackage.org) para descobrir de onde se baixa a JTA, com comando **rpm**, e descobrimos que precisamos procurar em http://java.sun.com/products/jta/.

4. Desta vez, tivemos que baixar dois ZIPs: o de classes e o de documentação. E copiamos ambos para o diretórios de fontes de RPM

```
bash# cd /diretorio/onde/baixei/JTA
bash# cp jta*-classes.zip jta*-doc.zip /usr/src/redhat/SOURCES
```

5. Construimos os pacotes finais e instalamos os RPMs gerados:

```
bash# cd /diretorio/onde/baixei/nosrc.rpm
bash# rpmbuild --rebuild jta*nosrc.rpm
bash# cd /usr/src/redhat/RPMS/noarch
bash# rpm -Uvh jta*rpm
```

   E a JTA está instalada.

4. Instalando Outros Softwares Java Pelo JPackage

Neste ponto, você já tem o repositório JPackage configurado no seu sistema, e a JVM de sua escolha instalada conforme ditam os padrões FHS de diretórios do Linux.

Agora é muito fácil instalar qualquer outra aplicação, biblioteca ou JAR disponível no JPackage, representado pelo nome do pacote na lista a esquerda em http://www.jpackage.org.

Para instalar ou atualizar um pacote, bastam os seguintes comandos respectivamente:

```
bash# yum install [nome do pacote]    # Para instalar.
bash# yum update [nome do pacote]     # Para atualizar.
```

O YUM, usando os metadados do repositório, vai resolver todas as dependências, baixar tudo que for necessário, e instalar os pacotes.

4.1. Exemplo: Instalando O Apache Tomcat

O Apache Tomcat é um servlet container, que se integra ao webserver e permite a criação e execução de aplicações web feitas em Java (servlets).

Para instalar o Tomcat, segundo nosso exemplo anterior, basta:

```
bash# yum install tomcat5
```

Após resolver todas as dependências, o YUM determinou que para instalar o Tomcat, seria necessário instalar também vários módulos do Jakarta, Axis, módulos de XML, etc. E tudo foi automaticamente baixado e instalado num mesmo passo.

5. Instalando O Eclipse

O Eclipse foi a princípio uma poderosa ferramenta de desenvolvimento de aplicações, ou IDE.

Desde a versão 3, ele foi reestruturado para ser um “servidor de aplicações” de desktop. Ou seja, se tornou o que chamamos de Rich Client Platform — ou RCP — que é uma base genérica que provê a infraestrutura padronizada que qualquer aplicação de desktop precisa. O IDE então passou a ser uma aplicação, um plugin, do RCP. O IDE Java está no JPackage com o nome de **eclipse-jdt**, e para instala-lo, basta:

```
bash# yum install eclipse-jdt
```

Como sempre, todos os outros módulos necessário para estes componentes serão automaticamente selecionados e instalados.

O ícone do Eclipse deve aparecer no menu inicial, pronto para ser usado.