Sumário

Objetivo:....2
Conjunto de softwares....2
Pacotes para GNU/Linux—Debian (.deb)....3
Estrutura de diretórios, scripts de distribuição e arquivos de configuração:...3
Processo de geração dos pacotes....5
Fases para geração/atualização....6
Teste de uso do software....6
Atualização dos projetos Maven....6
Publicação....8
Pacotes para GNU/Linux— RPM....8
Geração dos pacotes....8
Pré-requisitos....8
Procedimentos....8
1Arquitetura i386 (32 bits)....8
2Arquitetura x86_64 (64 bits)....9
3Independente de Arquitetura (noarch)....9
4Assinatura dos pacotes....10
Geração do repositório....10
Pré-requisitos....10
Procedimentos....10
Instaladores para MS-Windows®....11

Objetivo:

Neste manual você encontrará todas as informações necessárias para fazer a geração de pacotes de instalação dos softwares necessárias para o desenvolvimento com o Framework Demoiselle.

Podem haver 2 contextos para uso deste documento.
1 – Geração de toda a infraestrutura: Quando há necessidade de refazer os repositórios ou criar uma estrutura própria paralela à oficial.
2 – Atualização de algum(s) software(s) específico(s).

Atualmente há dois tipos de plataformas de sistemas operacionais suportadas pela comunidade: Sistemas do tipo Gnu/Linux e algumas versões do Microsoft Windows®. Para a plataforma Gnu/Linux são suportados os formatos de empacotamento do tipo Debian (.deb) padrão de distribuições como Debian, Ubuntu, Mint, por exemplo e do tipo RPM que é padrão das distribuições como Fedora, OpenSuse, Mandriva.

Para os procedimentos de instalação dos pacotes gerados, a documentação é a do Manual do Usuário disponível no site: http://demoiselle.sourceforge.net/infra/

Conjunto de softwares

Os softwares que fazem parte do projeto Demoiselle-Infra são aqueles necessários ou acessórios para o desenvolvimento de aplicações com o Framework Demoiselle.
Atualmente estão divididos em três subconjuntos:

1. IDE:
   Estarão neste conjunto os softwares de interface para ambiente de desenvolvimento. O software padrão da comunidade Demoiselle é o Eclipse (http://www.eclipse.org).
2. SERVER:
   Estarão neste conjunto os softwares que são os servidores de aplicação testados pela equipe de desenvolvimento do Demoiselle. Os servidores padrões são: Jboss (http://www.jboss.org) e o Tomcat (http://tomcat.apache.org)
3. TOOL:
   Estarão nesta categoria todos os softwares que servem de apoio ou são acessórios para o desenvolvimento. O Demoiselle-Nimble e o Demoiselle-Workspace são exemplos deste tipo de software.

Pacotes para GNU/Linux—Debian (.deb)

Estrutura de diretórios, scripts de distribuição e arquivos de configuração:

Este processo é baseado na documentação do Debian (https://www.debian.org/distrib/packages).

No repositório git do projeto: https://github.com/demoiselle/infra/tree/master/debian/InfraModelo há o modelo da estrutura de diretórios. Essa estrutura deve estar implantada no servidor Web que hospedará o repositório. A figura abaixo exemplifica a estrutura:

Na raiz do diretório estão os arquivos de scripts de distribuição:

- ***gerarepo_stable.sh*** : Arquivo de script para geração da versão ESTÁVEL (produção) dos pacotes. É necessário editá-lo para informar o número da chave GPG que fará a assinatura dos pacotes, e em alguns casos informar os caminhos dos diretórios que estão relativos.
- ***gerarepo_unstable.sh*** : Arquivo de script para geração da versão instável (Testes) dos pacotes. É necessário editá-lo para informar o número da chave GPG que fará a assinatura dos pacotes, e em alguns casos informar os caminhos dos diretórios que estão relativos.
- ***gpg-passphrase***: Conterá a senha para a chave GPG configurada nos arquivos acima, é necessário editá-lo para informar a senha.

gpg --import ComunidadeFrameworkDemoisellePub.asc
gpg --allow-secret-key-import --import ComunidadeFrameworkDemoisellePriv.asc

Há uma chave GPG criada para comunidade Demoiselle. Se a geração dos pacotes é para o repositório oficial é essa que deverá ser usada. Caso seja repositório local, pode ser usada qualquer outra chave.

Os outros diretórios na raiz são:
- cache: Este diretório é usado para armazenar os caches das gerações anteriores dos pacotes, isto evita que caso haja uma atualização pontual todo o repositório não seja reprocessado.
- conf: Contém os arquivos de configuração de distribuição para geração dos pacotes. Sendo eles:

- apt-demoiselle-ftparchive.conf: configuração para geração de pacotes na versão estável.
- apt-demoiselle-unstable-ftparchive.conf: configuração para geração de pacotes na versão instável (testes).
- apt-demoiselle-hardy.conf: configuração para geração de índices para versão/seção de codinome hary da distribuição ubuntu. (obsoleto, é substituído pelo universal).
- apt-demoiselle-lucid.conf: configuração para geração de índices para versão/seção de codinome lucid da distribuição ubuntu. (obsoleto, é substituído pelo universal).
- apt-demoiselle-precise.conf: configuração para geração de índices para versão/seção de codinome precise da distribuição ubuntu. (obsoleto, é substituído pelo universal).
- apt-demoiselle-raring.conf: configuração para geração de índices para versão/seção de codinome raring da distribuição ubuntu. (obsoleto, é substituído pelo universal).
- **apt-demoiselle-universal.conf:** configuração para geração de índices para seção universal. Por não depender da versão do sistema operacional foi definida essa seção. Essa é a configuração definitiva dos pacotes

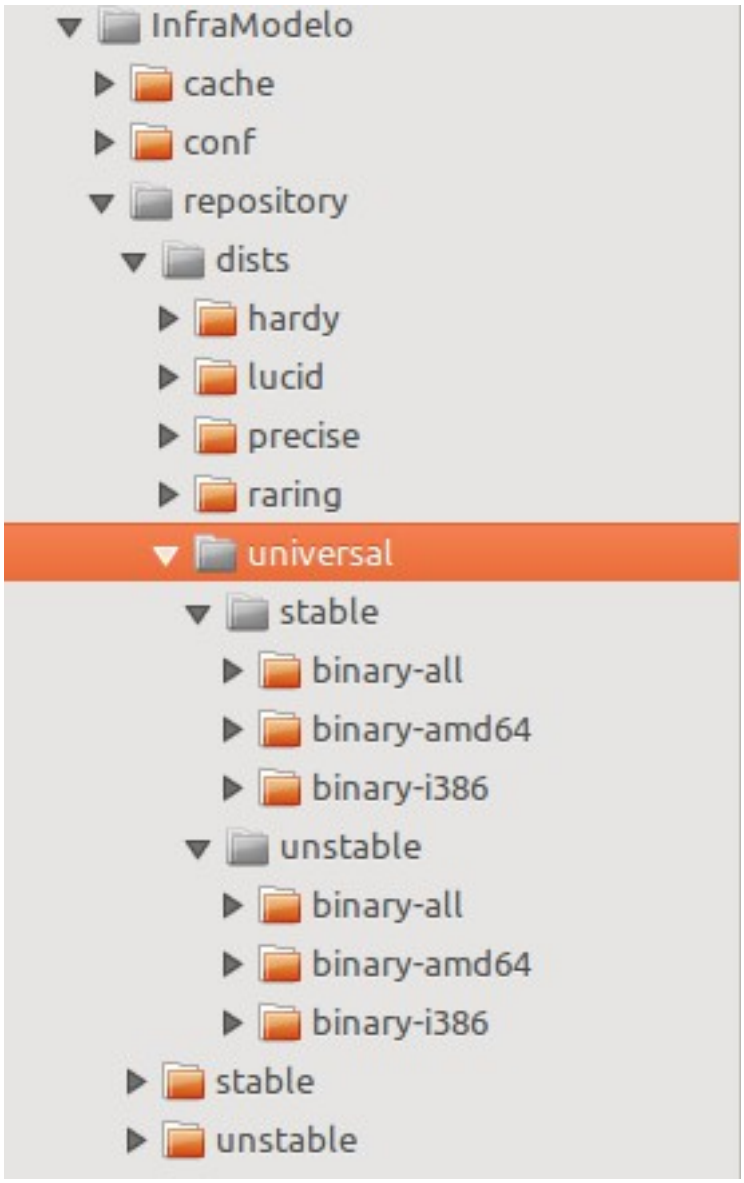
Os arquivos de configuração para versões específicas do ubuntu foram mantidas por questão de histórico.

Os arquivos e diretórios acima, NÃO devem ficar acessíveis publicamente no servidor, se for o caso devem estar em locais separados do diretório abaixo.

- repository:
Esta estrutura de diretórios é que estará com acesso público, pois será o repositório propriamente dito.
Contém outros 3 (três) subdiretórios:

- stable: É onde estarão os pacotes no formato .deb na versão estável.
- unstable: É onde estarão estar os pacotes no formato .deb na versão instável (teste).
- Dists: Contém mais 5 (cinco) subdiretórios. Quatro deles são específicos para versões antigas do ubuntu e não são mais necessárias e foram mantidos por questão de histórico. O diretório que importa é o chamado **universal**.
  - No diretório universal (assim como nos outros) está a estrutura de diretórios para as versões **stable** e **unstable** e para cada

uma delas as divisões por arquitetura de equipamento. (all, amd64 e i386). Conforme a figura abaixo.



Para processar os scripts de geração, essa estrutura é necessária.

Processo de geração dos pacotes

Toda a tarefa de geração dos pacotes (.deb), envio para o repositório e execução dos scritps de distribuição está baseada em projetos gerenciados pelo Apache Maven: https://github.com/demoiselle/infra/tree/master/debian/architectures
Esses scripts foram reformulados para que se tornassem de uso amplo da comunidade. Até a versão 1.3.0 (https://github.com/demoiselle/infra/tree/1.3.0) não havia documentação e o processo estava equivocadamente “amarrado” aos procedimentos internos do fundador da comunidade (SERPRO).

Os projetos estão divididos em três categorias:

1. **all**

Estarão nesta categoria todos os softwares que são independentes de arquitetura do sistema operacional (32 ou 64 bits).

2. amd64
   Estarão nesta categoria todos os softwares que são específicos para arquitetura 64 bits.
3. i386
   Estarão nesta categoria todos os softwares que são específicos para arquitetura 32 bits. Optou-se pela categoria i386 pois abrange desde os equipamentos mais antigos.

Essas categorias são previstas pelo modelo de pacotes debian.

Cada um dos projetos em cada categoria possui um arquivo chamado leiame.txt, neste arquivo estão as orientações para atualização de cada software já utilizado no projeto.

Caso haja necessidade de inclusão de um novo software, deve ser usado como base um software que esteja na mesma categoria.

Fases para geração/atualização

Teste de uso do software

O primeiro passo antes da geração ou atualização de qualquer software consiste em baixar (download) o software e testá-lo para certificar-se que está pronto para o uso pelo desenvolvedor e atende as necessidades do Framework Demoiselle.
Caso o software possua distribuição para mais de uma arquitetura (32 ou 64 bits) os testes devem ser feito em cada uma delas. Recomendamos usar o software VirtualBox (https://www.virtualbox.org/) que permite que sejam instalados/virtualizados vários sistemas operacionais em um mesmo equipamento.
Somente após este passo é que o software estará pronto para ser distribuído.

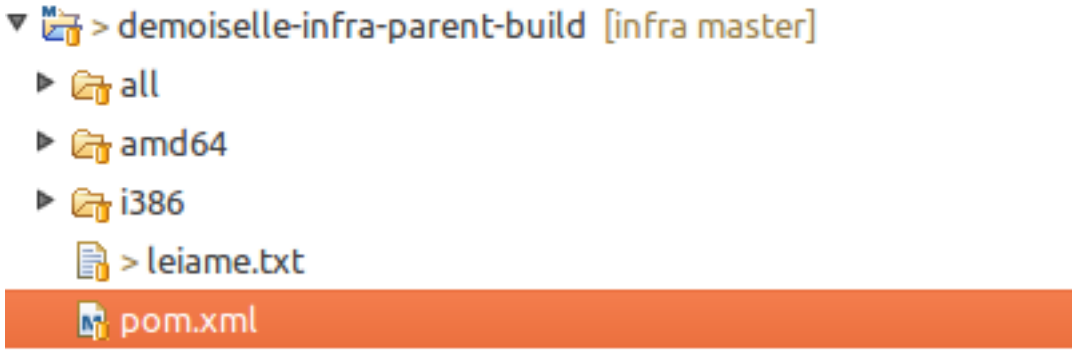
Atualização dos projetos Maven

Para atualizar os projetos é necessário o uso do Apache-Maven, que pode ser a distribuição desktop (https://maven.apache.org/download.cgi) ou o plugin do Eclipse (https://maven.apache.org/eclipse-plugin.html). Recomendamos o uso dos pacotes do próprio Demoiselle-Infra para esta atividade.
Com os softwares instalados basta baixar os fontes do repositório: https://github.com/demoiselle/infra.git
Se for utilizar o Eclipse há esse documento com um tutorial de como clonar os projetos: http://sourceforge.net/projects/demoiselle/files/tutoriais/guia-demoiselle-nimble-com-eclipse-github-groovy.pdf
Na raiz do projeto que será atualizado, leia atentamente o arquivo leiame.txt que fornece as instruções para atualização.

Como todo projeto gerenciado pelo Apache-Maven, a base são os arquivos POM.XML.

Os arquivos do tipo POM.XML possuem a característica de herança, ou seja o projeto pode herdar informações de outro arquivo POM.XML. Desta forma foi criado um arquivo onde estão centralizadas as informações de segurança e local onde os arquivos serão distribuidos.
Esse arquivo está localizado em /infra/debian/parent/pom.xml

No Eclipse estará no projeto: demoiselle-infra-parent-build, conforme mostra a figura abaixo:



As propriedades que devem ser atualizadas são as seguintes:

```
<sshUser>usuario</sshUser>
<sshKeyFile>./.ssh/demoinfra_rsa</sshKeyFile>
<sshPassword>senha</sshPassword>
<sshHost>end_host</sshHost>
<sshDirCopyTo>diretorio_remoto(unstable/stable)</sshDirCopyTo>
<sshCommand>/home/usu_remoto/repository/gerarepo_unstable.sh</sshCommand>
```

Exemplo:

```
<sshUser>demoiselle</sshUser>
<sshKeyFile>/home/demoiselle/.ssh/chave_rsa</sshKeyFile>
<sshPassword>demoiselle</sshPassword>
<sshHost>localhost</sshHost>
<sshDirCopyTo>/var/www/repository/unstable/</sshDirCopyTo>
<sshCommand>/home/demoiselle/repository/gerarepo_unstable.sh</sshCommand>
```

Comece sempre configurando os dados da versão Unstable (Testes).
Serão essas configurações que determinarão se serão gerados os pacotes estáveis ou de testes. E também as regras de segurança.

Com os dados atualizados, basta executar a fase “package” do Maven que os pacotes serão gerados, enviados para o servidor e no servidor os scripts de publicação serão executados.

Publicação.

Ao executar o empacotamento do(s) projeto(s) a publicação é feita automaticamente. Por convenção, a publicação é feita inicialmente em ambiente de testes (unstable). Após essa publicação, deve ser feita a divulgação na lista de usuários para os mesmos testem os pacotes.
Depois de testados e aprovados, configure as informações do projeto demoiselle-infra-parent-build para versão estável (stable) e execute novamente o empacotamento, feito isso os pacotes já estarão disponíveis para comunidade.

Pacotes para GNU/Linux— RPM

Distribuições como OpenSuse, Fedora, Mandriva, etc, utilizam o formato .rpm para distribuição de pacotes.
Para atender essas distribuições foram criados alguns processos e scripts para geração dos pacotes e de repositórios.

Os arquivos necessários para criação dos pacotes e repositório estão nesta área do GitHub: https://github.com/demoiselle/infra/tree/master/rpm

Geração dos pacotes

Pré-requisitos

Para geração dos pacotes em formato RPM, é necessário um ambiente (instalação) com uma distribuição que utilize este formato (ex: OpenSuse, Fedora). E também é preciso que a instalação corresponda à arquitetura a ser gerada (64 bits ou 32 bits)

É necessário que esteja instalado o pacote: ***RPM-BUILD***.

Clonar o repositório GIT https://github.com/demoiselle/infra/tree/master/rpm.

Procedimentos

Ao clonar o repositório haverão os seguintes diretórios:

git/infra/rpm/i386

git/infra/rpm/noarch

git/infra/rpm/x86_64

Cada diretório contém os arquivos necessários para geração dos pacotes para cada tipo de arquitetura.

1 Arquitetura i386 (32 bits)

O diretório git/infra/i386 (clonado do gitHub) conterá um arquivo de script (dir2rpm.sh) para geração dos pacotes para geração dos pacotes com

softwares que possuem versões que dependem da arquitetura do sistema operacional. E um arquivo de especificação de um metapacote para indicar as dependências para o ambiente para a arquitetura 32 bits (demoiselle-2-infra_32.spec).

Neste diretório também estarão os arquivos:

- leiame_demoiselle-2-infra.txt

- leiame_eclipse.txt

- leiame_workspace.txt

Cada um destes arquivos contém as instruções para geração dos respectivos pacotes (.rpm), leia cada um deles e gere os pacotes.

2 Arquitetura x86_64 (64 bits)

O diretório git/infra/x86_64 (clonado do gitHub) conterá um arquivo de script (dir2rpm.sh) para geração dos pacotes com softwares que possuem versões que dependem da arquitetura do sistema operacional. E um arquivo de especificação de um metapacote para indicar as dependências para o ambiente para a arquitetura 64 bits (demoiselle-2-infra_64.spec).

Neste diretório também estarão os arquivos:

- leiame_demoiselle-2-infra.txt

- leiame_eclipse.txt

- leiame_workspace.txt

Cada um destes arquivos contém as instruções para geração dos respectivos pacotes (.rpm), leia cada um deles e gere os pacotes.

3 Independente de Arquitetura (noarch)

O diretório git/infra/ noarch (clonado do gitHub) conterá um arquivo de script (dir2rpm_noarch.sh) para geração dos pacotes com softwares que não dependem da arquitetura do sistema operacional.

Neste diretório também estarão os arquivos:

- leiame_cassandra.txt

- leiame_ireport.txt

- leiame_jboss.txt

- leiame_maven.txt

- leiame_maven-repo.txt

- leiame_mongo.txt

- leiame_nimble.txt

- leiame_soapui.txt

- leiame_squirrel.txt

- leiame_tomcat.txt

- leiame_wildfly.txt

Cada um destes arquivos contém as instruções para geração dos respectivos pacotes (.rpm), leia cada um deles e gere os pacotes.

4 Assinatura dos pacotes

Após a geração de todos os pacotes, é preciso assiná-los digitalmente. A assinatura dos pacotes é a garantia da autoria e sem assinatura o processo de instalação gerará mensagens de alerta que podem confundir o usuário.

Para assinar os pacotes gerado é preciso de um par de chaves do tipo RSA (pode ser criada com o comando: *ssh-keygen*). No caso da comunidade Demoiselle há uma chave padrão de posse dos patrocinadores.

Os procedimentos para assinatura são os seguintes:

- Criar ou importar a chave privada.
  Para importar a chave privada da comunidade o comando é:
  gpg --import ComunidadeFrameworkDemoisellePriv.asc
  Para verificar se a chave foi importada com êxito o comando é:
  gpg –list-keys
  Para importar a chave pública da comunidade o comando é:
  sudo rpm --import ComunidadeFrameworkDemoisellePub.asc
- Copiar o arquivo git/infra/rpm/ponto_rpmmacros para o diretório home do usuário.
- Editar o arquivo ponto_rpmmacros alterando as propriedades %_gpg_path e %_gpg_name (este último somente se não for usada a chave da comunidade).
- Renomear o arquivo para .rpmmacros
- Assinar os pacotes.
  Para assinar todos os pacotes num diretório o comando é:
  rpm --addsign *.rpm
  Para assinar individualmente cada pacote o comando é:
  rpm --addsign <nome pacote>
  ex: rpm --addsign demoiselle-2-infra-3-1.0-1.0.i586.rpm

Geração do repositório

Para facilitar o procedimento de instalação, pelo usuário final, dos pacotes gerados o recurso recomendado é a criação de um repositório. Assim não é necessário que o usuário faça download e instalação dos pacotes de forma manual. Além disso, o processo de atualização também pode ser automatizado.

Pré-requisitos

O repositório pode ser criado em qualquer distribuição Linux com um servidor Web (ex: apache) e o pacote: createrepo

Procedimentos

Os passos para criação (ou atualização) deste tipo de repositório são:

- Criar os diretórios para cada arquitetura, no servidor web.

  Exemplo:

  mkdir -pv /var/www/rpm/i386/demoiselle

  mkdir -pv /var/www/rpm/x86_64/demoiselle

  mkdir -pv /var/www/rpm/noarch/demoiselle

- Copiar os pacotes gerados de acordo com a arquitetura nos diretórios que foram criados no passo anterior.
- Executar o comando de geração de índices, para cada diretório.

  Exemplo:

  createrepo /var/www/rpm/i386/demoiselle

  createrepo /var/www/rpm/x86_64/demoiselle

  createrepo /var/www/rpm/noarch/demoiselle

- Assinar os índices dos repositórios, com o programa GPG. A sintaxe é gpg -a -u <número da chave> --detach-sign <diretorio/nome do arquivo a ser assinado>

  Exemplo:

  gpg -a -u B32B7170 --detach-sign /var/www/rpm/i386/demoiselle/repodata/repomd.xml

  gpg -a -u B32B7170 --detach-sign /var/www/rpm/x86_64/demoiselle/repodata/repomd.xml

  gpg -a -u B32B7170 --detach-sign /var/www/rpm/noarch/demoiselle/repodata/repomd.xml

- É importante exportar a chave pública para cada repositório. A sintaxe é: gpg -a –exprt [número da chave] > diretorio/nome do arquivo

  gpg -a --export B32B7170 > /var/www/rpm/i386/demoiselle/repodata/repomd.xml.key

  gpg -a --export B32B7170 > /var/www/rpm/x86_64/demoiselle/repodata/repomd.xml.key

  gpg -a --export B32B7170 > /var/www/rpm/noarch/demoiselle/repodata/repomd.xml.key

Executados esses passos o repositório estará pronto para ser utilizado. As instruções para uso do repositório para instalação dos programas está no guia do usuário no site do projeto.

Instaladores para MS-Windows®.

Processo a ser remodelado.