



RPM パッケージ作成ハンズオン 入門編

—LILO 2011.07 Monthly Seminar, 第6回 Vine Linux ユーザの集い—

山本 宗宏 <munepi@vinelinux.org>

2011 年 7 月 17 日@株式会社グッデイ

produced by Project Vine and exciting guys

概 要

RPM パッケージになっていないサンプルソフトウェアを通して、RPM パッケージの作成を体験するセミナーです。入門編では、RPM パッケージ作成に関わるパッケージのビルドやテスト、vbuilder を使った src.rpm のビルドテストなどの一連の作業を学べます。

入門編の流れ

1. 準備
2. サンプルソフトウェアのコンパイル
3. 雛形 spec ファイルの編集
4. パッケージのビルド
5. パッケージのインストール, アンインストール
6. パッケージのビルドテスト

1 準備

手元に Vine Linux 5 または Vine Linux 6 がインストールされた計算機を用意して下さい。このレジュメでは、Vine Linux 6 i386 を前提として解説しています。もし ppc, x86_64 を用意している方は、適時読み替えてコマンドを実行して下さい。

1.1 ~/.rpmmacros の編集

RPM パッケージを作成するための最低限の設定をしましょう。適当なテキストエディタを用いて、`~/.rpmmacros` を開いて下さい。

Vine Linux 6

```
1 # %packager Your Name    <your mail address>
2 # %packager      Munehiro Yamamoto <munepi@vinelinux.org>
```

```

3 %packager      munepi
4
5 # GnuPG によるパッケージ署名を行う
6 # %_signature  gpg
7 # %_gpg_name Your Name <your mail address>
8
9 # ソースファイル置き場をパッケージ別にする
10 %_sourcedir %{_topdir}/SOURCES/%{name}

```

Vine Linux 5

```

1 %_topdir ${HOME}/rpm
2
3 # %packager Your Name <your mail address>
4 # %packager      Munehiro Yamamoto <munepi@vinelinux.org>
5 %packager      munepi
6
7 # gpg signing
8 # %_signature  gpg
9 # %_gpg_name Your Name <your mail address>
10
11 # more useful macros
12 %_sourcedir      %{_topdir}/SOURCES/%{name}

```

- `%packager` : あなたの ID (半角英数字の文字列) を入力して下さい。あなたのハンドルネームやニックネームで構いません。特に、そのような ID をお持ちでない方や分からない方は、上記の例を参考に、名前とメールアドレスを入力して下さい。
- 今回は作成していただいた RPM パッケージに GPG 署名をしません。
- `%_sourcedir %{_topdir}/SOURCES/%{name}` : パッケージごとにソースやパッチなどの格納ディレクトリを設けます。パッケージごとにディレクトリを分けておくと、複数のパッケージをメンテナンスするときに、それぞれのファイルたちが混ざりません。デフォルトでは、`%_sourcedir %{_topdir}/SOURCES` となっています。

1.2 sudo の設定

`sudo` は一般ユーザに特定のコマンドを `root` 権限で実行させるためのコマンドです。 `su` コマンドから `root` になってもよいのですが、今回は積極的に `sudo` を使います。

Vine Linux 6

あなたはおそらく手元の計算機の管理者です。念のために、`id` コマンドを実行して、`wheel` グループに所属しているか確認して下さい。

```

1 $ id
2 uid=1000(munepi) gid=1000(munepi) groups=1000(munepi),10(wheel)

```

Vine Linux 5

あなたはおそらく手元の計算機の管理者です。sudo をインストールします。

```
1 $ su -c "apt-get install sudo"
```

あなたのアカウントを sudoers に追加しましょう。

```
1 $ su -c /usr/sbin/visudo
```

vi の操作に不馴れな方は、環境変数 EDITOR に gedit などの使いなれたテキストエディタを代入して、以下を実行して下さい。

```
1 $ EDITOR=gedit su -c /usr/sbin/visudo
```

以下の例を参考にしながら、“## Allow root to run any commands anywhere” と書かれた行の下に、自分のアカウントを追加して保存します。

```
1 ## Allow root to run any commands anywhere
2 root    ALL=(ALL)        ALL
3 munepi  ALL=(ALL)        ALL
```

2 パッケージングしてみよう

2.1 サンプルを動かす

今回の RPM パッケージハンズオンのために、echowlvl というサンプルソフトウェアを用意しました。このサンプルソフトウェアを RPM パッケージにしてみましょう。

2.1.1 サンプルをダウンロード

wget コマンドを用いて、echowlvl-0.1.tar.gz をダウンロードします。

```
1 $ wget http://trac.vinelinux.org/repos/people/munepi/documents/11/
   echowlvl-0.1.tar.gz
```

2.1.2 サンプルをコンパイル

ソフトウェアをビルドするための最小限の開発環境を整えます。

```
1 $ sudo apt-get install build-essential
```

先ほどダウンロードしたサンプル echowlvl-0.1.tar.gz を展開して、“いつもの呪文”（./configure && make && make install のこと）を唱えてみましょう。

```
1 $ tar xvf echowlvl-0.1.tar.gz
2 $ cd echowlvl-0.1
3 $ ./configure
4 $ make
5 $ ./echowlvl
```

コンパイルしてできた実行ファイル `echowlvl` を実行すると、どうなったでしょうか？ みなさんで一斉に叫んでみましょう！

2.2 spec ファイルを書く

`echowlvl` が“いつもの呪文”を唱えてビルドできたので、RPM パッケージを作成するためのソースコードともいえるべき spec ファイルを書きましょう。

2.2.1 雛形 spec ファイルのダウンロード

`echowlvl` の RPM パッケージ作成にすぐさまとりかかれるように、spec ファイルの雛形を用意しました。wget コマンドを用いて、`echowlvl-v1.spec` をダウンロードして下さい。

```
1 $ wget http://trac.vinlinux.org/repos/people/munepi/documents/11/
    echowlvl-v1.spec
```

ダウンロードした `echowlvl-v1.spec` を `~/rpm/SPECS/` へ移動します。

```
1 $ mv echowlvl-v1.spec ~/rpm/SPECS/
```

2.2.2 spec ファイルの構成

spec ファイルは主に 4 つの部分から構成されています。

1. パッケージ情報
2. スクリプト
3. ファイルリスト
4. 更新履歴

各部を順番に見ながら埋めていきましょう。

■ **パッケージ情報** パッケージの基本情報はすべてここに凝縮されています。特にパッケージ依存関係は、rpm や apt に正しい情報を伝えるために重要になります。

```
1 ## -*- coding: utf-8-unix -*-
2 ## This is a template spec file for LIL0 2011.07 Monthly Seminar.
3 Name:                echowlvl
4 Version:             0.1
5 Release:             1%{?_dist_release}
6 Summary:             Sample program for RPM making hands-on (lms201107)
7 Summary(ja):        RPM パッケージ作成ハンズオン (lms201107) 用サンプルプログラム
8 Group:              Applications/Other
9 License:            GPLv3+
10 URL:               http://trac.vinlinux.org/wiki/OfflineMeeting/20110711
11 Source0:           http://trac.vinlinux.org/repos/people/munepi/documents
                       /11/{name}-%{version}.tar.gz
12
13 BuildRoot:         %{_tmppath}/%{name}-%{version}-%{release}-root
```

```

14
15 Distribution:    YOUR LINUX DISTRIBUTION
16 Vendor:         YOUR VENDOR
17 Packager:       YOUR NAME (or YOUR ID)
18
19 %description
20 echowlvl is a sample program for RPM making hands-on (lms201107).
21
22 %description -l ja
23 echowlvlは RPM パッケージ作成ハンズオン (lms201107) 用サンプルプログラム
    です。

```

- Name: パッケージ名です。 `%{name}` というマクロで参照できます。今回は、 `echowlvl` として下さい。
- Version: ソフトウェアのバージョンです。 `%{version}` というマクロで参照できます。今回は、 `0.1` として下さい。
- Release: リリース番号と Vine Linux のバージョンを組み合わせで書きます。リリース番号は、同じバージョンで何回目のリリースになるかを意味します。 `%{?_dist_release}` は、Vine Linux のバージョンに置き換えられます。例えば、Vine Linux 6 の場合は、 `1%{?_dist_release}` は `1v16` となります。 `%{release}` というマクロで参照できます。
- Summary, Summary(ja): パッケージの概要です。パッケージの概要をそれぞれ英語と日本語で記入して下さい。
- Group: ソフトウェアの種類です。今回は `Applications/Other` にします。
- License: ソフトウェアのライセンスです。今回は `GPLv3+` にします。
- `%description, %description -l ja`: Summary よりも詳しいパッケージの解説を書きます。

■ **スクリプト** ソフトウェアのソースアーカイブを展開し、必要であればパッチを当て、コンパイル、インストールの一連の作業を記述します。

```

1 %prep
2 %setup -q
3
4 %build
5 %configure
6 %__make %{?_smp_mflags}
7
8 %install
9 %__rm -rf $RPM_BUILD_ROOT
10 %__make install DESTDIR=$RPM_BUILD_ROOT
11
12 %clean
13 %__rm -rf $RPM_BUILD_ROOT

```

“いつもの呪文”をざっくり spec ファイルに記述すると、このようになります。 `echowlvl` の rpm 化はこれで十分なので、今回はこのままにしておきます。

■ **ファイルリスト** 実際にインストールされるファイルリストを書きます。

```

1 %files
2 %defattr(-,root,root,-)

```

さて、何を書けばよいのでしょうか？ 一旦、保留にしておきます。

■**更新履歴** 更新履歴は、パッケージのメンテナだけでなく、パッケージに興味のある人が見るかもしれません。他の人が見てよいように、できるだけ分かりやすく記述するように心掛けましょう。

あなたの名前とメールアドレスを埋めて下さい。

```

1 %changelog
2 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.1-1
3 - first release
4
5 ## end of file

```

2.3 パッケージのビルド

さきほどダウンロードした echowlvl-0.1.tar.gz を ~/rpm/SOURCES/echowlvl に移します。

```

1 $ mkdir ~/rpm/SOURCES/echowlvl
2 $ mv echowlvl-0.1.tar.gz ~/rpm/SOURCES/echowlvl/

```

では、rpmbuild -ba でパッケージをビルドしてみます。

```

1 $ rpmbuild -ba ~/rpm/SPECS/echowlvl-v1.spec

```

そういえば、%files のファイルリストを埋めていませんでした。早速、/usr/bin/echowlvl をファイルリストに追加しましょう。

```

1 %files
2 %defattr(-,root,root,-)
3 %doc AUTHORS COPYING ChangeLog INSTALL NEWS README
4 %{_bindir}/echowlvl

```

%doc に入れたドキュメントファイルは、/usr/share/doc/%{name}-%{version} に格納されます。

再び、rpmbuild -ba で echowlvl パッケージをビルドして下さい。今度は echowlvl パッケージができあがっていませんか？

2.4 パッケージのインストール

echowlvl パッケージができたら、インストールをします。echowlvl パッケージが期待した通りにインストールできて、動作することを確認します。

```

1 $ sudo apt-get install ~/rpm/RPMS/i386/echowlvl-0.1-1v16.i386.rpm

```

echowlvl コマンドを実行して動作確認をしましょう。先ほどを同じ挙動をしているでしょうか？

2.5 パッケージのアンインストール

今度は、echowlvl パッケージをアンインストールしてみます。

```
1 $ sudo apt-get remove echowlvl
```

echowlvl コマンドを実行できてしまったりしないか、確認してみましょう。

```
1 $ echowlvl
2 bash: echowlvl: コマンドが見つかりません
3 $ which echowlvl
4 /usr/bin/which: no echowlvl in (/home/munepi/bin:/usr/local/bin:/usr/bin
   :/bin:/usr/local/sbin:/usr/sbin:/sbin)
```

3 さらにパッケージングしてみよう

3.1 サンプルが更新されたようです！

wget コマンドを用いて、echowlvl-0.2.tar.gz をダウンロードして下さい。

```
1 $ wget http://trac.vinlinux.org/repos/people/munepi/documents/11/
   echowlvl-0.2.tar.gz
```

新しくなった echowlvl をコンパイルして、動作することを確認して下さい。

3.1.1 ビルドに必要なライブラリを追加

もし echowlvl をコンパイルできなかった方は、エラーメッセージを見ながら、原因を追究してみましょう。

今回は、curses.h というヘッダファイルがコンパイルに必要なだったようです。/usr/include/curses.h は、ncurses-devel パッケージに格納されています。apt-get を使って ncurses-devel パッケージをインストールします。

```
1 $ sudo apt-get install ncurses-devel
```

3.1.2 spec ファイルを更新

echowlvl の新しいバージョンに合わせて、echowlvl-vl.spec を編集しましょう。Version はバージョンを 0.2 にして下さい。%changelog にソフトウェアが更新されたことを記します。

```
1 ## -*- coding: utf-8-unix -*-
2 ## This is a template spec file for LIL0 2011.07 Monthly Seminar.
3 Name:          echowlvl
4 Version:       0.2
5 Release:       1%{_dist_release}
6 ...
7
8 %changelog
9 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.2-1
```

```

10 - new upstream release
11
12 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.1-1
13 - first release
14
15 ## end of file

```

3.1.3 パッケージを更新

rpmbuild -ba でパッケージのビルドを試みて下さい。新しいバージョンに対応した echowlvl パッケージはできましたでしょうか？ echowlvl パッケージができた場合は、パッケージのインストール、テスト、アンインストールもお忘れなく！

4 ビルドテストしましょう

パッケージができたなら、パッケージのビルドテストをしましょう。いつでもどんな環境でもパッケージングできた方が嬉しいですね。「あの人の環境ではビルドできるけど、私の環境ではビルドできひんよ？」、「私の環境でもビルドできたけど、この機能が使えへんよ？」といったことが起こると不都合ですよ。

パッケージのビルドテストには、vbuilder を使います。vbuilder は、(chroot という) 子環境に Vine Linux の最小環境から構築し、パッケージの依存関係などのチェックをしてからパッケージのビルドするツールです。

vbbootstrap, vbuilder に関する詳細は、以下の URL に記載されています。

<http://trac.vinelinux.org/wiki/VineBootstrap>

4.1 vbuilder の基本操作

vbuilder は vbbootstrap パッケージに格納されています。

```
1 $ sudo apt-get install vbbootstrap
```

基本的な流れは以下の通りです。

```

1 $ sudo vbuilder clean      (過去に構築した子環境を削除)
2 $ sudo vbuilder build      (子環境に最小環境を構築)
3 $ sudo vbuilder build-rpm hoge-1.0-1v16.src.rpm

```

vbuilder にこれらのアクションを一度に渡すこともできて、以下のようにすれば、順次 clean → build → build-rpm のように実行します。

```
1 $ sudo vbuilder clean build build-rpm hoge-1.0-1v16.src.rpm
```

この場合、build を省略できます。

```
1 $ sudo vbuilder clean build-rpm hoge-1.0-1v16.src.rpm
```

なお、\$ sudo vbuilder clean build を実行すると、約 600MB 程度の容量を消費します。vbuilder を実行するには、ハードディスクの容量にご注意下さい。

4.2 src.rpm を vbuilder に投げてみる

echowlvl-0.2-1vl6.i386.rpm と同時に、もう 1 つ echowlvl-0.2-1vl6.src.rpm という rpm ができていました。src.rpm は、ソフトウェアのソースアーカイブやパッチファイルなど、spec ファイルをまとめたソースパッケージです。echowlvl-0.2-1vl6.src.rpm から echowlvl-0.2-1vl6.i386.rpm をビルドできます。

vbuilder に echowlvl-0.2-1vl6.src.rpm を投げてみましょう。一度 vbuilder を実行すると、一連の作業が終わるまでしばらくかかります。果たして vbuilder は echowlvl-0.2-1vl6.i386.rpm を返してくれるのでしょうか？

```
1 $ sudo vbuilder clean build-rpm ~/rpm/SRPMS/echowlvl-0.2-1vl6.src.rpm
```

4.3 依存関係を解決する

vbuilder に echowlvl-0.2-1vl6.src.rpm を投げたら、“curses.h がありません”と言われてしまいました。echowlvl-0.2-1vl6.src.rpm はソースパッケージとして不完全のようです。

/usr/include/curses.h は ncurses-devel パッケージに格納されています。ncurses-devel パッケージを事前にインストールしていると、echowlvl-0.2-1vl6.src.rpm がビルドできそうです。早速、試してみましょう。

```
1 $ sudo vbuilder clean install-rpm ncurses-devel build-rpm ~/rpm/SRPMS/
   echowlvl-0.2-1vl6.src.rpm
```

4.3.1 spec ファイルを更新

echowlvl-vl.spec を編集しましょう。Release はリリース番号を 2 にして下さい。BuildRequires: ncurses-devel を追加します。%changelog に BuildRequires: ncurses-devel を追加したことを記します。

```
1 ## -*- coding: utf-8-unix -*-
2 ## This is a template spec file for LILLO 2011.07 Monthly Seminar.
3 Name:                echowlvl
4 Version:             0.2
5 Release:             2%{?_dist_release}
6 ...
7 Source0:             http://trac.vinelinux.org/repos/people/munepi/documents
                        /11/%{name}-%{version}.tar.gz
8
9 BuildRequires:       ncurses-devel
10
11 BuildRoot:           %{_tmppath}/%{name}-%{version}-%{release}-root
12 ...
13
14 %changelog
15 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.2-2
16 - added BuildRequires: ncurses-devel
17
18 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.2-1
```

```
19 - new upstream release
20
21 * Sun Jul 17 2011 YOUR NAME <YOUR E-MAIL ADDRESS> 0.1-1
22 - first release
23
24 ## end of file
```

4.3.2 パッケージのビルドテスト

再び, `rpmbuild -ba` で `echowlvl` パッケージをビルドして下さい. `echowlvl-0.2-2vl6.i386.rpm`, `echowlvl-0.2-2vl6.src.rpm` が生成されるはずです.

今度は, `vbuilder` に `echowlvl-0.2-2vl6.src.rpm` を投げてみましょう. `echowlvl-0.2-2vl6.i386.rpm` を返してくれたら, パッケージのビルドテストに成功です.

```
1 $ sudo vbuilder clean build-rpm ~/rpm/SRPMS/echowlvl-0.2-2vl6.src.rpm
```

`vbuilder` が返してくれた `echowlvl-0.2-2vl6.i386.rpm` のパッケージをテストして下さい. パッケージのインストール, アンインストールも問題なければ, 入門編は終了です. お疲れ様でした.