Source-Build

概要

本稿は応用課題として提示する.
EmacsとVimをソースコードからビルドして，導入する.
root権限を使わない導入方法を検討する.

目次

ソースビルド

今までは，アプリケーションは， `yum` というパッケージマネージャーを用いてインストールしていた.
これは，公開されているリポジトリから，アプリケーションのバイナリを入手してインストールする.
しかし，パッケージマネージャーの利点は，どちらかというと依存関係の解決やアプリケーションの一元管理にあり，リポジトリとの連携は副次的なものである.

旧来，パッケージマネージャーがなかった時代のアプリケーションのインストールは，バイナリをどこからか入手したり，ソースコードを入手し，手元でコンパイルし，依存関係を手動で解決していた.
つまり，1つのアプリケーションをインストールしようとすると，依存するアプリケーションをインストールしたり，アプリケーションに合わせてアップグレードやダウングレードする必要があった.
さらに，場合によっては，依存関係のアプリケーションの依存関係の...というように延々と続いていくこともあった.

では，何故この時代になっても，ソースコードのビルドにてアプリケーションのインストールを行うのか.
それは，主に2つの理由である.

1つ目は，最新ないしは指定のバージョンのアプリケーションを求めるときである.
リポジトリに公開されるアプリケーションのパッケージは，様々なテスト (それこそ依存関係など)　をパスしてから公開される.
CentOSの場合は，資料 `Install-Linux.md` でも述べたとおり安定性を優先していたり，FedoraやRedHatが先にあることから，公開されるパッケージはやや古いものとなってしまう.
そのため，最新のアプリケーションを求めるときは，自ずとソースコードのビルドが必要となる.
近年では，Dockerを代表とするコンテナ技術の発展により，求めるアプリケーションを仮想環境内でビルドして実現する運用も考えられる.

2つ目は，自らの環境や運用に合わせてチューニングする場合である.
ビルド時に，バイナリのインストール先を指定する `prefix` の指定や，不必要な機能は除外してインストールすることができる.
例えば，CUIオンリーのサーバにインストールするときに，GUIの機能は無効化してインストール，である.

このとき，`configure` へのオプション指定で解決する人やソースファイル自体に自作のパッチを適応し，悪魔じみたチューニングを行うことができる．

一般的に，以下の流れでインストールされる．
しかし，色々な手法があるため，必ずアプリケーションの `README` ファイルないしは `INSTALL` ファイルを確認すること．

```
(1) $ ./configure
(2) $ make
    $ sudo make install
    $ make clean
```

(1) `./configure`
インストールを行うOS，CPUアーキテクチャ，コンパイラ(gccやclなど)，必要な依存関係を調査する．
依存関係が不足しているときは，エラーを出して終了する．
調査した情報を基に，環境に合わせたMakefileを生成する．
オプションにて細かく指定し，チューニングできる．

(2) `make`
ソースコードをコンパイルする．
`clean` を指定すると，生成したバイナリの掃除ができる．(`$ make clean`)
`install` を指定すると，生成したバイナリや関連するファイルを規定のディレクトリにコピーする．(`$ make install`)
ここで，上記の例示に `sudo` が付与されている理由を検討する．
一般的に`./configure`時の`prefix`による指定がないときは，バイナリのコピー先は`/usr/local`となる．
この領域は，一般ユーザは書き込むことができないためである．
逆に言うと，バイナリのコピー先を一般ユーザが書き込める領域に指定すると，`sudo`を用いずにコピー(インストール)することができる．
良くある領域としては，`/home/user/usr/local`などである．
即ち，`/usr/local`を模して，自分のホームディレクトリにインストールすることである．

==========
余談であるが，ビルドとコンパイルは明確に異なる．
コンパイルはソースファイルからバイナリ(機械語)への翻訳であり，ビルドは関連ファイルのリンクを含めた作業である．
==========

準備

`yum` を使って，以下のパッケージをインストールする．

- `gcc` … C/C++のコンパイルに使う．他にFortranやGoもコンパイルできる．
- `make` … Makefileに記述された依存関係に従って，コマンドを実行する．コンパイルを楽にするために使われる事が多い．
- `ncurses-devel` … 端末に依存しないように画面を更新する．`-devel`と付くパッケージは，開発に必要なライブラリやヘッダファイルを含んでいる．

- `wget` ... ファイルのダウンロードに使用する．機能が豊富であり，応用するとWebクローラーを作れる．
- `bzip2` ... bz2ファイルの解凍に使用する．今回は，Vimのパッケージの展開のみに使用する．

Emacs

1. `wget` を使って，http://ftp.jaist.ac.jp/pub/GNU/emacs/emacs-27.1.tar.gz より圧縮ファイルをダウンロードする．
2. `tar` コマンドを使い展開する．
3. `cd` コマンドを使い，`emacs-27.1` ディレクトリに移動する．
4. `./configure` には， `--without-all` オプションをつけて上記の流れを実行する．

Vim

1. `wget` を使って，https://ftp.nluug.nl/pub/vim/unix/vim-8.2.tar.bz2 より圧縮ファイルをダウンロードする．
2. `tar` コマンドを使い展開する．
3. `cd` コマンドを使い，`vim82` ディレクトリに移動する．
4. `./configure` には， `--disable-gui` と `--without-x` オプションをつけて上記の流れを実行する．

応用課題

書式

条件通りに実行できていることを，文章や実行結果，図で示せ．
例えば， `ls -R` や `diff`，`history`によるコマンド実行履歴を使う．

以下の課題は，EmacsもしくはVimのどちらかだけで構わない．
どちらか一方が分かればもう一方も同様の手順でできるため，両方やっても手間にはならない．

課題1

上記の手順で実際にインストールしたエディタのバージョンを示せ．
もし，特別に気に入ってるバージョンがあれば，特筆し，そのバージョンでも良い．
`--version` オプションで確認できる．

課題2 (Keyword: prefix，configure，option)

エディタのインストール時に，root権限を発動している．
これは何故であるか検討し，root権限を使用しない方法でインストールし，実行せよ．
なお，このとき上記でインストールしたエディタは，`$ sudo make uninstall` でアンインストールすることを強く推奨する．

課題3 (Keyword: 環境変数，PATH，which)

応用1で導入したエディタを実行するとき，バイナリを絶対パスで指定しなければならない．
通常のコマンドのように，ファイル名だけを指定すれば起動するようにせよ．

kilo

1000行で実装されているエディタとして， **kilo** というものがある．
極力，他のライブラリを使わず，C言語で実装されている．
ソースコードが公開されており，`make` でコンパイルするとすぐに使える．
シンタックスハイライト，ショートカットキー，画面の更新に対応しており，必要最低限の機能を有している．
参考文献を下記に貼り付けているので，学習や遊戯にいかがだろうか．

参考文献

- パッケージのテスト https://vinelinux.org/docs/vine6/developers-guide/tester.html , 2020/11/26
- ./configure;make;make installにはどんな意味がある？ - ITmedia エンタープライズ https://www.itmedia.co.jp/help/tips/linux/l0302.html , 2020/11/25
- ncurses(3) manページ https://nxmnpg.lemoda.net/ja/3/ncurses , 2020/11/25
- GNU Emacs - GNU Project https://www.gnu.org/software/emacs/ , 2020/11/25
- welcome home : vim online https://www.vim.org/ , 2020/11/25
- vim-jp » Vimのユーザーと開発者を結ぶコミュニティサイト https://vim-jp.org/ , 2020/11/25
- antirez/kilo: A text editor in less than 1000 LOC with syntax highlight and search. https://github.com/antirez/kilo, 2020/11/25
- Build Your Own Text Editor https://viewsourcecode.org/snaptoken/kilo/index.html , 2020/11/25
- C言語1000行以下のエディタ「Kilo」を理解する(1) シンプルな内部構造 | マイナビニュース https://news.mynavi.jp/article/kilo-1/ , 2020/11/25

奥付


```
Name     竹原 一駿 ( Ichitoshi TAKEHARA )
所属     香川大学大学院 工学研究科 信頼性情報システム工学専攻 最所研究室 M1
メールアドレス   itakehara@fw.ipsj.or.jp
```



```
--------------------------------------------------------------------------------
2020/12/03 初版
2021/05/25 2版
--------------------------------------------------------------------------------
```