RPMS for fun and Profit

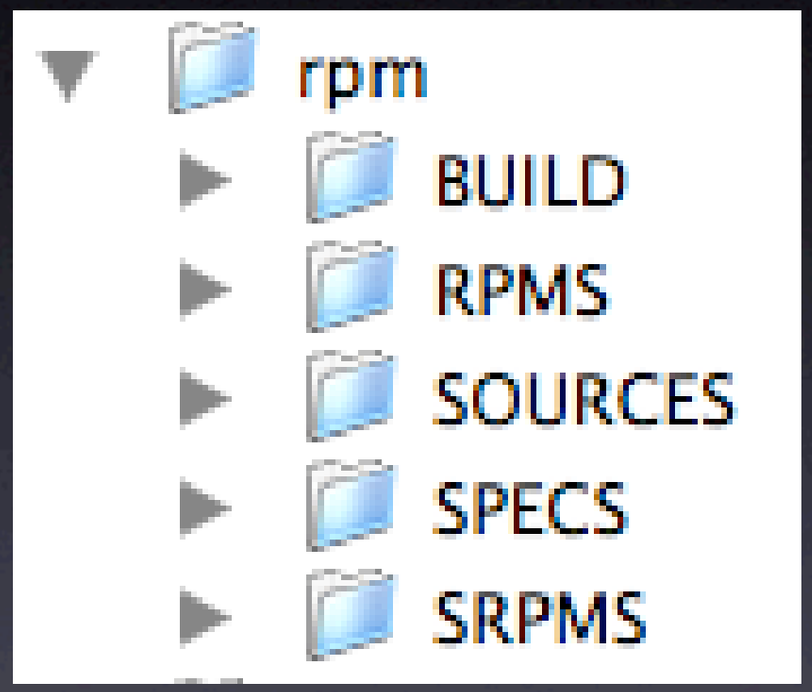
or, How I Learned to Love the Package

First things first

- Don't build packages as root, setup your user account to build packages.

- Setup a directory structure as illustrated on the right.

- Configure your .rpmmacros file. This tells rpmbuild where to build.




```
%packager  T.R. Fullhart <kayos@genetikayos.com>
%_topdir   /home/kayos/rpm
```

Ingredients

- 1 cup original source distribution.
- 2 tsp. .spec file.
- Patches to taste
- Put the source distribution and the patches in the SOURCES directory. The .spec file goes in the SPECS directory.

Anatomy of a .spec file

- Headers
- Prep
- Build
- Install
- Files
- Changlog

```
Summary:    GNU Hello
Name:       hello
Version:    2.1.1
Release:    1
Source:     http://ftp.gnu.org/gnu/hello/hello-2.1.1.tar.gz
URL:        http://www.gnu.org/software/hello/hello.html
License:    GPL
Group:      System Environment/Libraries

%description
This project is an example that demonstrates GNU coding standards.

%prep
%setup

%build
CFLAGS="$RPM_OPT_FLAGS" ./configure --prefix=%{_prefix}
make

%install
rm -rf $RPM_BUILD_ROOT
make DESTDIR=$RPM_BUILD_ROOT install

%clean
rm -rf $RPM_BUILD_ROOT

%files
%defattr(-,root,root)
%doc AUTHORS BUGS ChangeLog COPYING INSTALL NEWS README THANKS TODO
%{_prefix}/bin/hello
%{_prefix}/info/hello.info
%{_prefix}/lib/charset.alias
%{_prefix}/man/man1/*
%{_prefix}/share/locale/*/LC_MESSAGES/hello.mo
%{_prefix}/share/locale/locale.alias

%changelog
* Sat Oct 4 2003 T.R. Fullhart <kayos@genetikayos.com>
- First draft of the spec file
```

Headers

```
Summary:    GNU Hello
Name:       hello
Version:    2.1.1
Release:    1
Source:     http://ftp.gnu.org/gnu/hello/hello-2.1.1.tar.gz
URL:        http://www.gnu.org/software/hello/hello.html
License:    GPL
Group:      Applications
%description
This project is an example that demonstrates GNU coding standards.
```

- Meta-data of the package.
- Most of these are required.
- Can also specify what requirements this package has.

The Prep Stage

```
%prep
%setup
```

- %prep is the start of the prep section.
- In %prep, you unpack your sources and apply your patches.
- %setup is a macro that knows how to unpack common distributions (.tar, .tgz, .zip)
- There is also a %patch macro that makes it easy to apply patches.
- This section is run through /bin/sh, so you can use bourne shell commands. %setup and %patch are just macros for a block of shell script.

The Build Stage

```
%build
CFLAGS="$RPM_OPT_FLAGS" ./configure --prefix=%{_prefix}
make
```

- In %build, you configure and do your build, usually with make.
- This section is run in /bin/sh so you can use all of your bourne shell commands.
- $RPM_OPT_FLAGS is an environment variable set by rpmbuild to turn on flags such as optimization.
- %{_prefix} is a macro to where the package should be installed. By default, it's /usr.
- $RPM_OPT_FLAGS and %{_prefix} can be configured in your .rpmmacros file.

The Install Stage

```
%install
rm -rf $RPM_BUILD_ROOT
make DESTDIR=$RPM_BUILD_ROOT install
```

- %install is sometimes tricky.
- The trick is that you need to get the software installed in $RPM_BUILD_ROOT, not the real root of the system.
- $RPM_BUILD_ROOT is an environment variable set by rpmbuild. It's usually set to a dir in /var/tmp/, but it's configurable.
- ./configure type build systems usually support DESTDIR for installing in a different root than what was configured.

The Package Stage

```
%files
%defattr(-,root,root)
%doc AUTHORS BUGS ChangeLog COPYING INSTALL NEWS README THANKS TODO
%{_prefix}/bin/hello
%{_prefix}/info/hello.info
%{_prefix}/lib/charset.alias
%{_prefix}/man/man1/*
%{_prefix}/share/locale/*/LC_MESSAGES/hello.mo
%{_prefix}/share/locale/locale.alias
```

- %files is a list of files that should be in the package. It's impossible for rpmbuild to automatically figure out what files are in the package, you have to tell it.
- %defattr() is a macro to define the default permissions and owner of the files when they are installed.
- %doc lists the files (relative to the build dir) that should be put in /usr/share/doc/packagename.
- You can use wildcards and you can specify the directory if everything in the directory should go into the package.
- I'm reusing the %{_prefix} macro from before. Since I did it in both places, I can do things like redefining it to /usr/local or /opt if I wanted to rebuild the package to install in those places.

Building the package

```
[kayos@genetikayos.com rpm]$ rpmbuild -bb SPECS/hello.spec

... bunch of stuff during %prep stage like unpacking...

... bunch of stuff during %build stage like running ./configure and make ...

... bunch of stuff during %install stage like running install ...

... some packaging information like the provides and requires stuff...
Finding  Provides: (using /usr/lib/rpm/find-provides)...
Finding  Requires: (using /usr/lib/rpm/find-requires)...
Provides: mailreader
PreReq: /bin/sh /bin/sh rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
Requires(interp): /bin/sh /bin/sh
Requires(rpmlib): rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
Requires(post): /bin/sh
Requires(preun): /bin/sh
Requires: ld-linux.so.2 libc.so.6 libc.so.6(GLIBC_2.0) libc.so.6(GLIBC_2.1.3)
Wrote: /home/kayos/rpm/RPMS/i386/hello-2.1.1-1.i386.rpm

... clean up

[kayos@genetikayos.com rpm]$
```

Tricks of the Trade

- Get it to build and install the normal way before you even attempt to make an RPM. You don't want to be debugging the build, install, and packaging at the same time.
- Steal code. What many people do is get the .spec files out of SRPMS and use those as a template. Most .spec files for perl modules and libraries are exactly the same.

Stuff I didn't cover

- There are a lot of things you can do with macros. Reading example .spec files can show you some neat tricks.
- You can have one .spec file that generates multiple packages.
- You can digitally sign your packages so someone can't tamper with the package.
- I covered an easy example, some types of packages are difficult.

Where to go from here

- Packaging Software with RPM
  http://www-106.ibm.com/developerworks/library/l-rpm1/
- RPM HOWTO: RPM at Idle
  http://en.tldp.org/HOWTO/RPM-HOWTO/index.html
- Maximum RPM
  http://www.rpm.org/max-rpm/
- The fight, rpm package building introduction
  http://freshrpms.net/docs/fight/
- Example spec files
- My RPM Tutorial