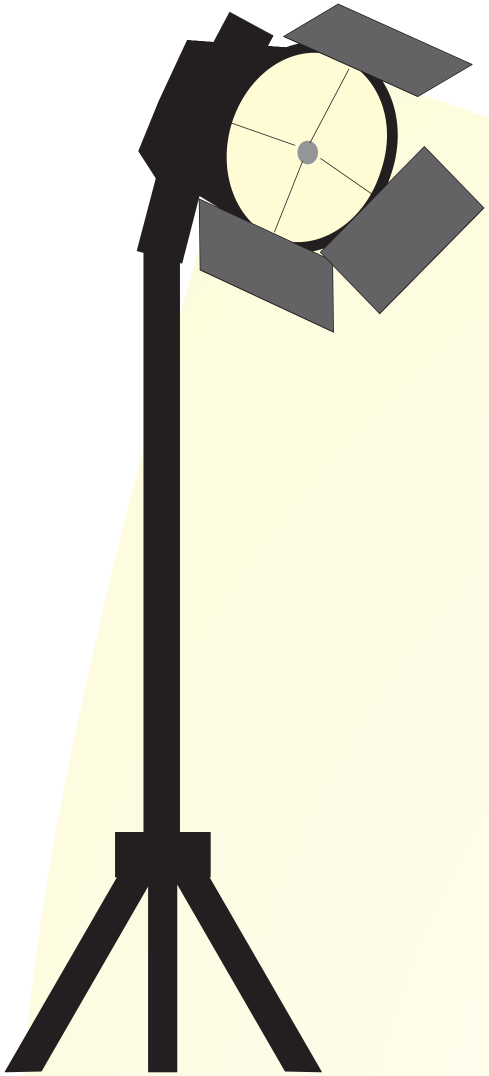


Building packages with *RPM*™

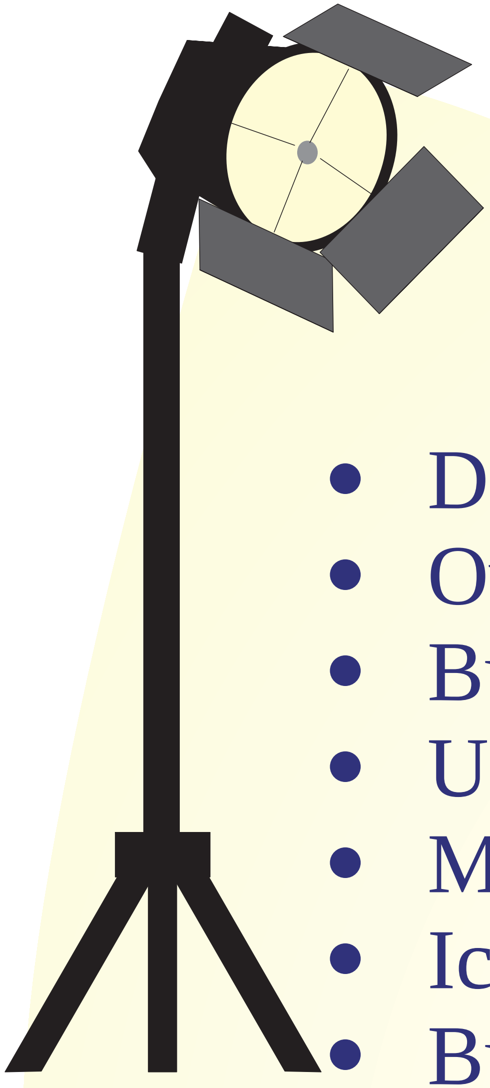
by Erik Troan

Red Hat Software

Getting RPM

- RPM included with:
  - Red Hat Linux 2.x
  - Red Hat Linux 3.0.3
  - CND 1.0
- ftp://ftp.redhat.com/pub/code/rpm

http://www.redhat.com/redhat/products/rpm



Overview

- Design goals of RPM
- Overview of .spec files
- Building a simple package
- Using macros in .spec files
- Multiple architecture support
- Icons
- Building GNU readline

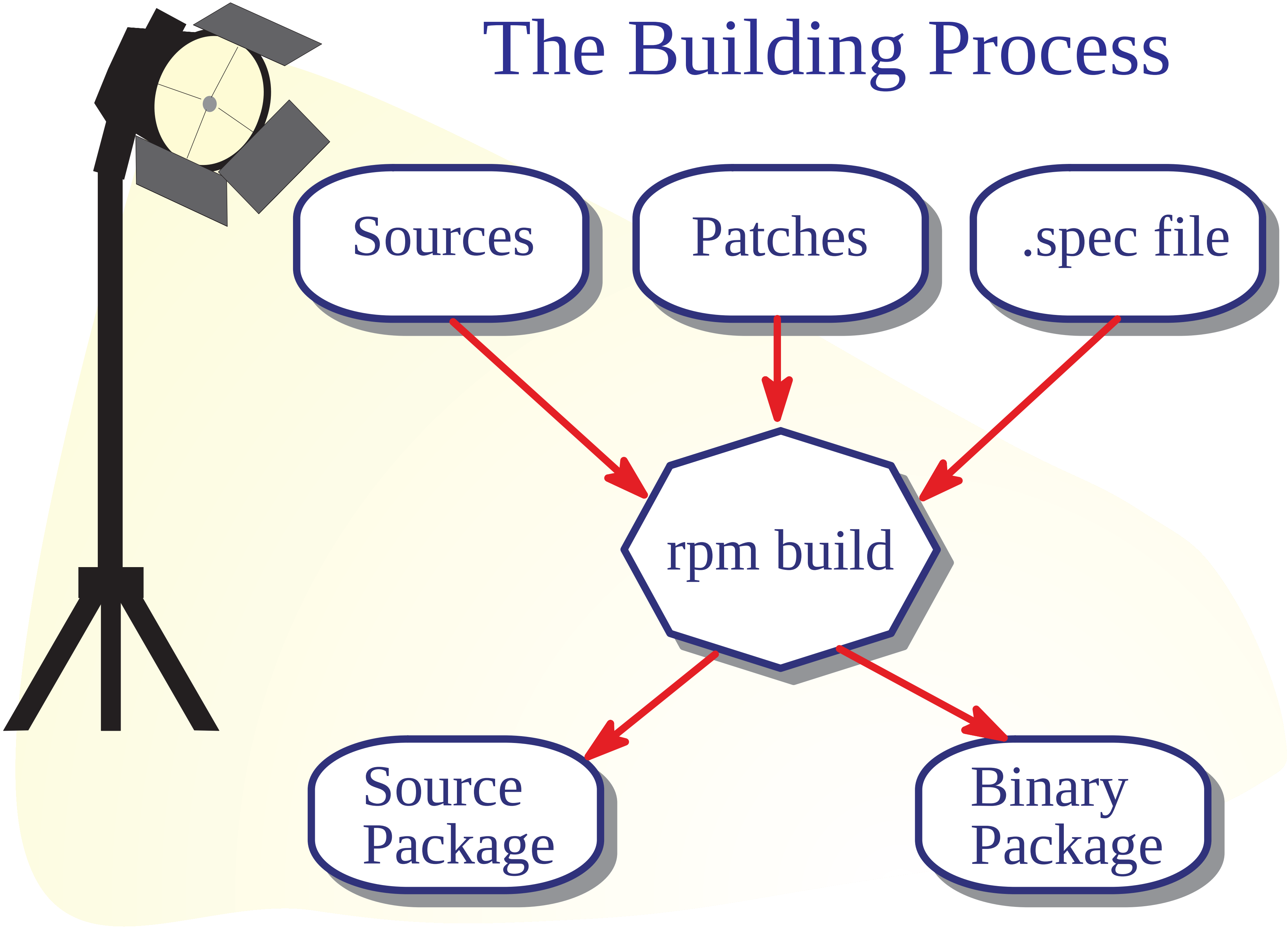
Design Goals

- Clean upgrading
- Verification
- Easy distribution of packages
- Easy to build packages
- Pristine sources

Pristine Sources

- Lets users know what changes the package builder had to make
- Makes it much easier to build packages for new versions of the program
- Makes it easy to "back out" changes

The Building Process



The .spec file

- source files
- patch files
- build instructions
- install instructions
- pre/post in/uninstall scripts
- file list and file flags
- subpackage specifications

All package information that isn't a source file or a patch file has to go into the .spec file!

Syntax of a .spec file

- keywords are prefaced by %
- exact syntax depends on section
  - header
    - <Tag>: <argument>
    - Some arguments can extend over multiple lines
  - shell script sections (build, install, etc)
    - normal shell commands
    - # for comments
    - %macro for RPM macros
  - file list
    - file names (with path)
    - globbing okay
    - %flag, %macro used

Simple .spec file

```
Description: locks virtual consoles
Name: vlock
Version: 0.8
Release: 3
Source:ftp://sunsite/pub/vlock-0.8.tar.gz
Copyright: GPL
Group: Utilities/Console

%prep
%setup

%build
make

%install
install -m 644 vlock.1 /usr/man/man1
install -m 755 -s vlock /usr/bin

%files
/usr/bin/vlock
/usr/man/man1/vlock.1
```

The .spec file
The Header

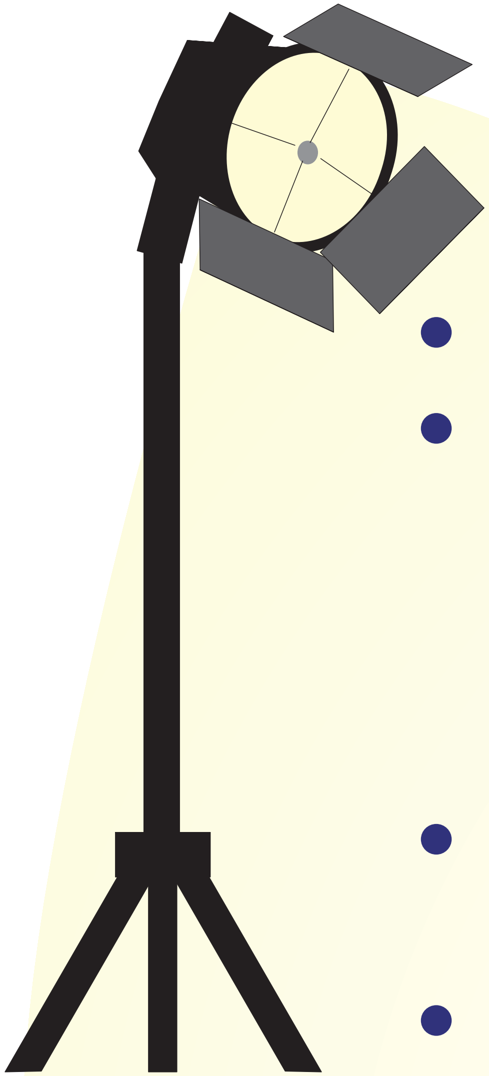
- *Description*
  - full description of this package
  - with RPM 2.0, *Summary*: should also be used to give a shorter description
- *Name*, *Version*, *Release*
- *Copyright*
  - should summarize the redistribution terms
  - free text, no format is enforced
- *Group*
- *Source* (URLs preferred)

The .spec file
Prep, Build, Install

- All of these sections are shell scripts
- *%prep* section
  - unpacks sources
  - applies patches
  - *%setup* and *%patch* macro makes this simple (normally)
- *%build*
  - generates makefiles (if necessary)
  - builds binaries
- *%install*
  - installs programs and documentation

The .spec file: The filelist

- specifies all of the files to include in the package
- directories are recursed, unless preceded by *%dir*
- filename globbing is fine
- *%config* marks config files
- *%doc* marks doc files (and places them in /usr/doc)
  - files in /usr/man, /usr/doc and /usr/info are always docs



%setup macro

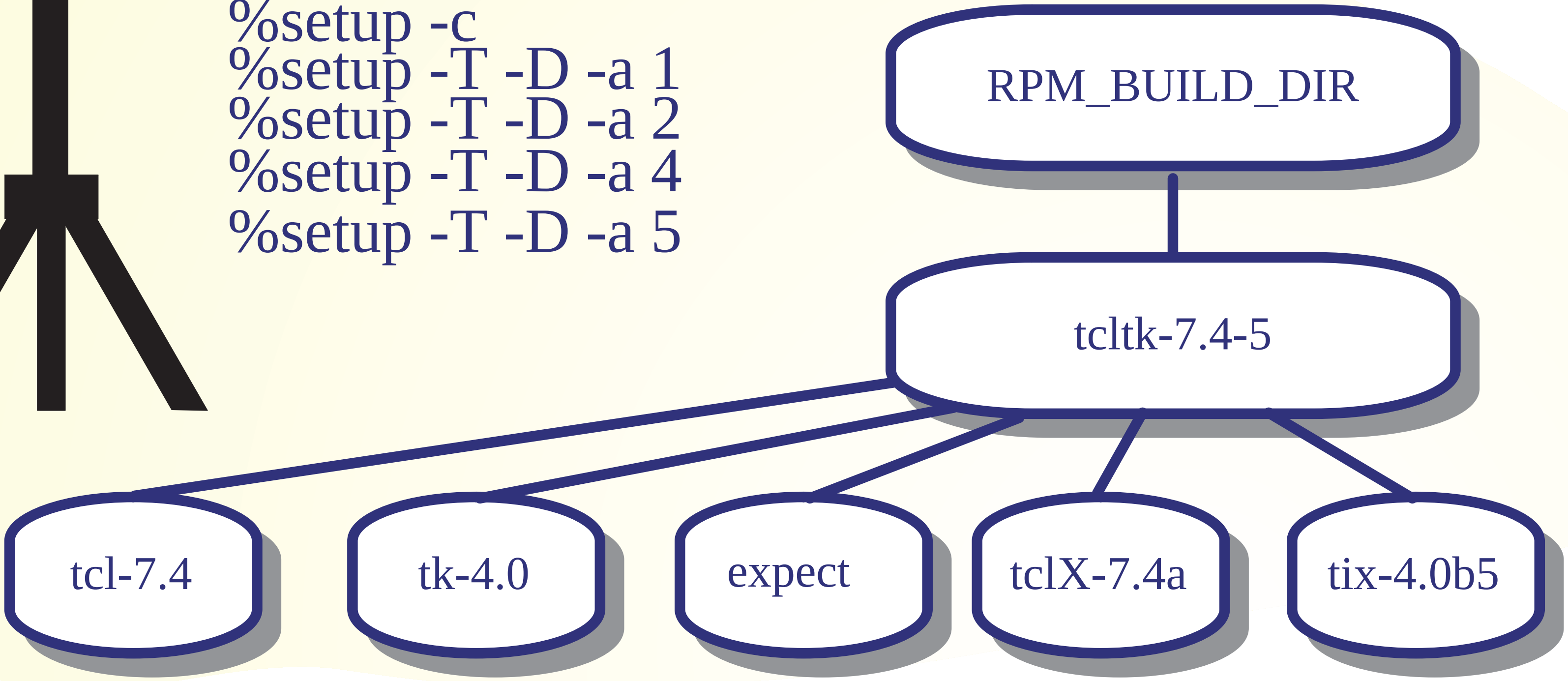
- Used in *%prep* section
- Normally, the default behavior is fine
  - removes the build directory (defaults to *<NAME-VERSION>*)
  - unpacks the first source file
  - cd's into the build directory
- *-n <dir>* flag is used to change the build directory
- *-c* creates and cd's into directory before unpacking the first source file
- "unpacking" handles compressed, gzipped, and normal tar files

%setup macro (cont'd)

- Other options are only needed for multiple sources or oddly packaged files
  - -*D* don't remove the build directory
  - -*T* don't unpack the first source package
  - -*a <num>* unpack source <num> after cd'ing into the build directory
  - -*b <num>* unpack source <num> before cd'ing into the build directory

%setup macro Example

Source0: ftp.cs.berkeley.com:/pub/tcl/tcl7.4.tar.Z
Source1: ftp.cs.berkeley.com:/pub/tcl/tk4.0.tar.Z
Source2: ftp.cme.nist.gov:/pub/expect/expect.tar.Z
Source4: ftp.neosoft.com:/pub/tcl/distrib/tclX7.4a-p1.tar.gz
Source5: ftp.cis.upenn.edu:/pub/ioi/Tix4.0b5.tar.gz
%setup -c
%setup -T -D -a 1
%setup -T -D -a 2
%setup -T -D -a 4
%setup -T -D -a 5



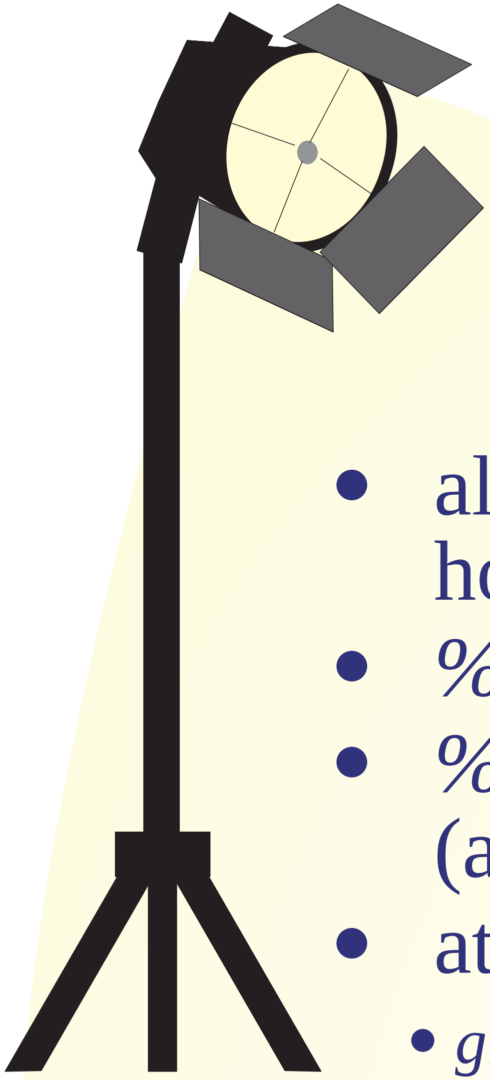
%patch macro

- applies patches via standard patch command
- applies first patch by default
- options
  - *-P <num>* applies patch number <num> (*%patch<num>* is a shortcut)
  - *-p<num>* strips <num> leading directories from the filenames in the patch
  - *-b <ext>* saves backup files with the extension <ext> (.orig is the default)

%doc, %docdir macros

- goes in the filelist (*%files* section), but it also installs files
- *%doc <path1> <path2>*
- does: cp -ar <path1> <path2> <docdir>
- <docdir> is /usr/doc/*name-ver-release*
- all files (and directories) in <docdir> get added to the filelist and marked as doc files
- *%docdir <path>* marks all files in <path> as doc files

%verify flag



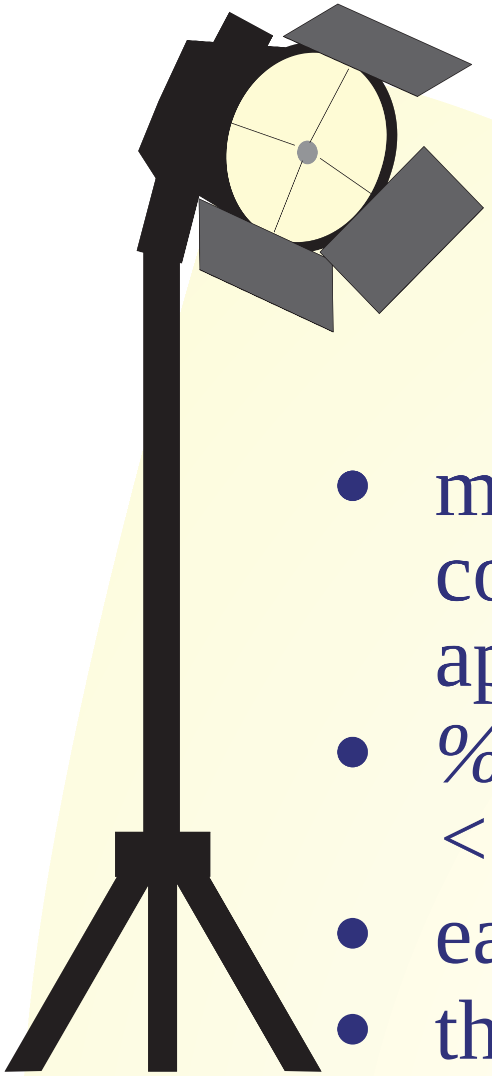
- allows the package builder to control how package verification works
- *%verify(attribs)* verifies only (attribs)
- *%verify(not attribs)* verifies just (attribs)
- attribs are:
  - *group* - GID
  - *mtime* - timestamp
  - *mode* - permission bits
  - *rdev* - major and minor
  - *user* - UID
  - *size* - file sizes
  - *md5* - file checksum
  - *link* - symlink value

Configuration Files

- configuration files are handled differently during upgrades and package removals
- marked by *%config* in the filelist (for example, *%config /etc/passwd*)
- RPM tries to keep old config files
- RPM will never replace a config file without making a backup copy
- RPM does not erase modified config files

Multiple Architecture Support

- *Exclude* and *Exclusive* tags in the header
  - control what platforms a package is built on
  - *Exclude: i386 axp* prevents the package from building on Intel or Alpha machines
  - *Exclusive: sparc* lets the package build only on SPARC's
- these let developers know what platforms a package should work on



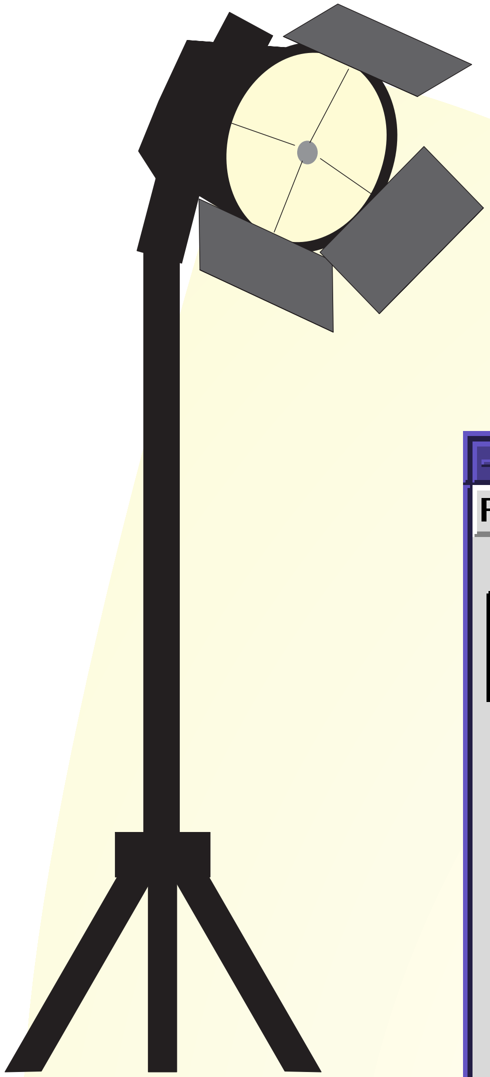
Multiple Architecture Support (cont'd)

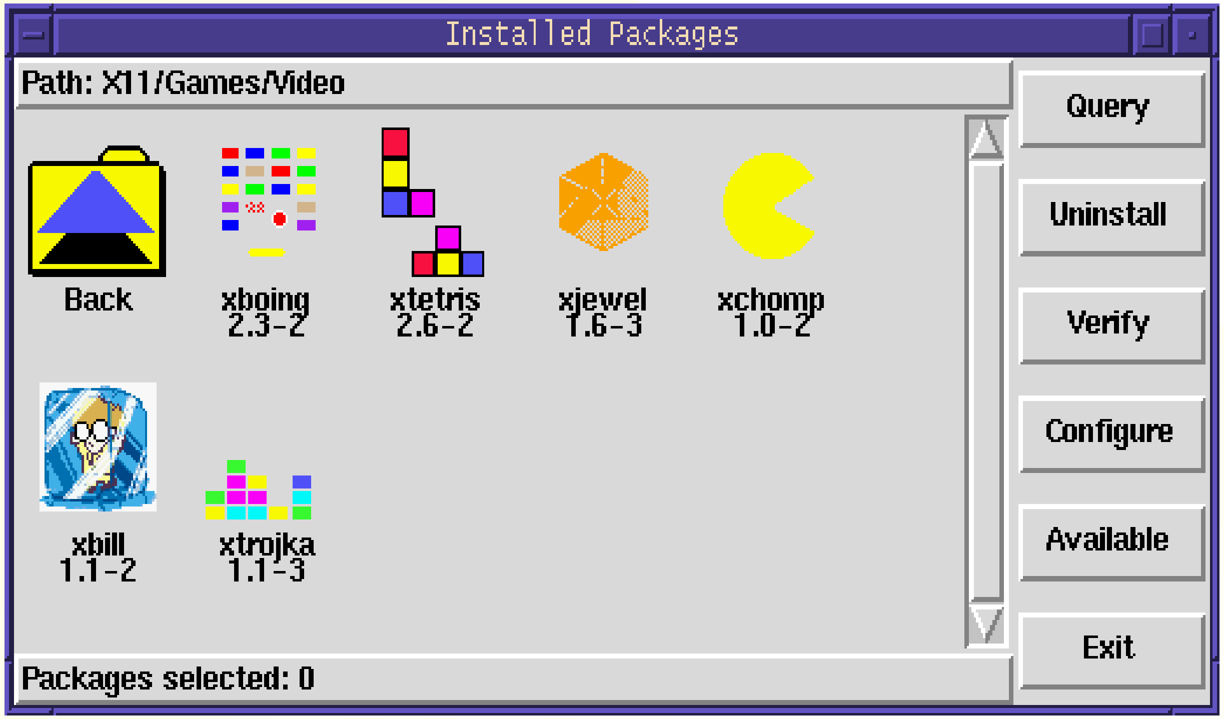
- macros similiar to C's preprocessor control what parts of the .spec file apply for each platform
- *%ifarch <archlist>*, *%ifnarch <archlist>*, and *%else*
- each *%if* must end with *%endif*
- these contructs let you
  - apply different patches
  - use different filelists
  - conditionally build shared libs

Package Icons

- makes glint look better
- RPM 1.x used .gif files exclusively
- RPM 2.x supports .gif but we prefer X Pixmap (.xpm) files
- specify them in the header with *Icon: <filename>*

Why use icons?

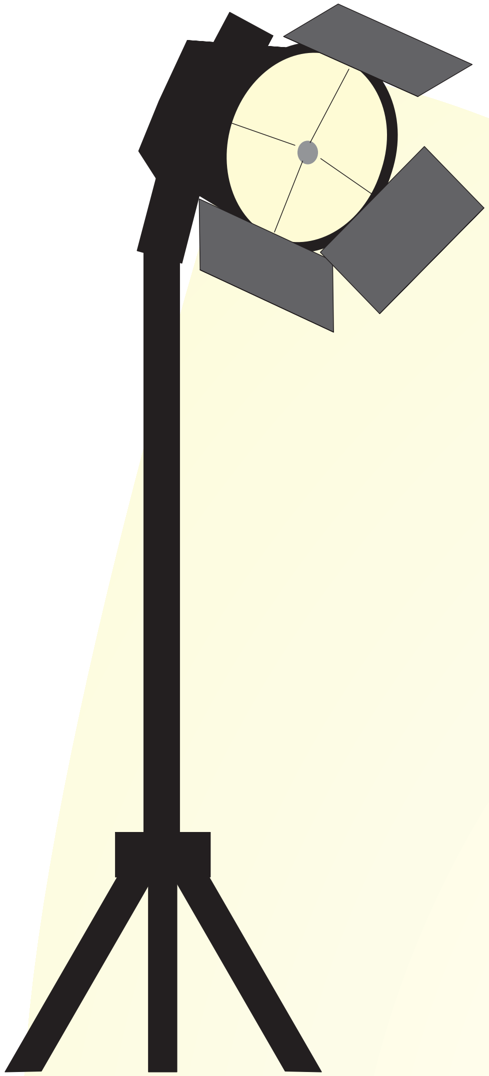




Building GNU Readline: The Header

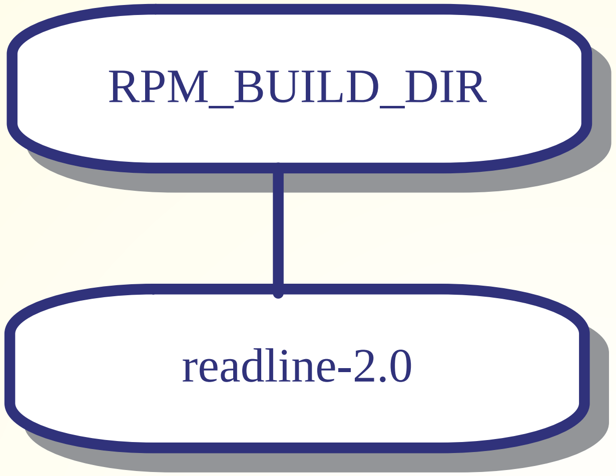
Description: lib for reading lines from the user
Name: readline
Version: 2.0
Release: 7
Copyright: GPL
Group: Libraries
Source: ftp://prep.ai.mit.edu:/pub/gnu/
readline-2.0.tar.gz
Patch0: readline-2.0-shlib.patch
Patch1: readline-2.0-alpha.patch

%package devel
Description: development files for readline
Group: Libraries

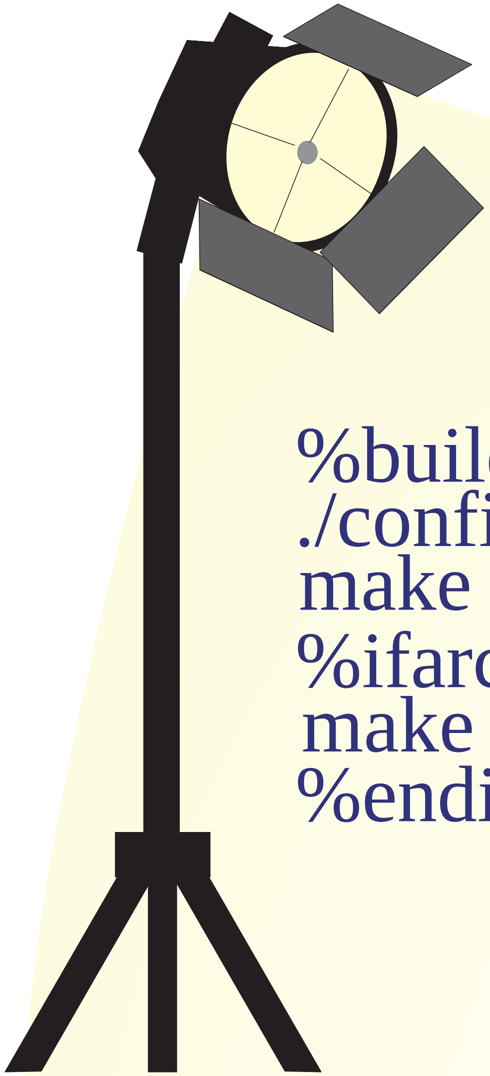


Building GNU Readline: %prep section

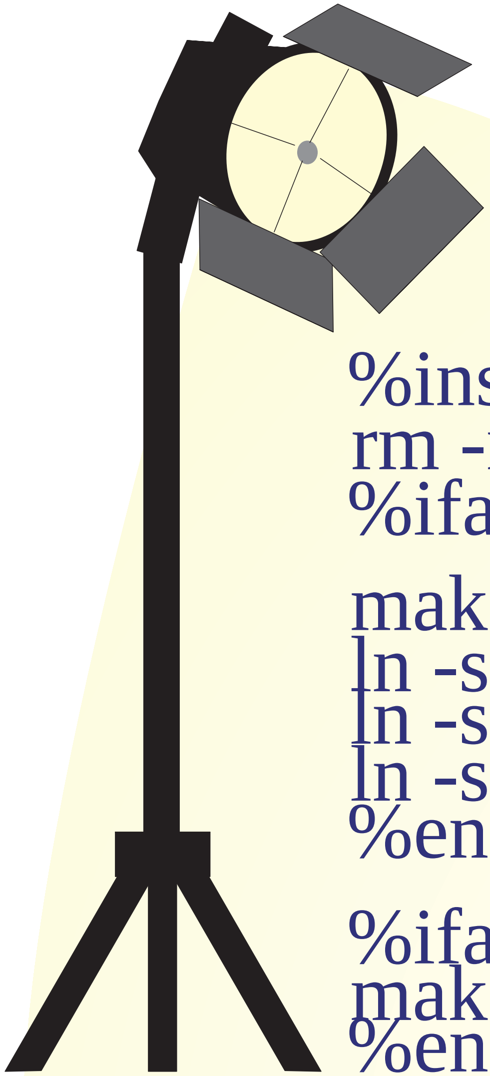
```
%prep
%setup
%patch -p1
%patch1 -p1
```



Building GNU Readline: %build section



```
%build
./configure --prefix=/usr
make CFLAGS="$RPM_OPT_FLAGS"
%ifarch i386
make CFLAGS="$RPM_OPT_FLAGS" shared
%endif
```



Building GNU Readline: %install section

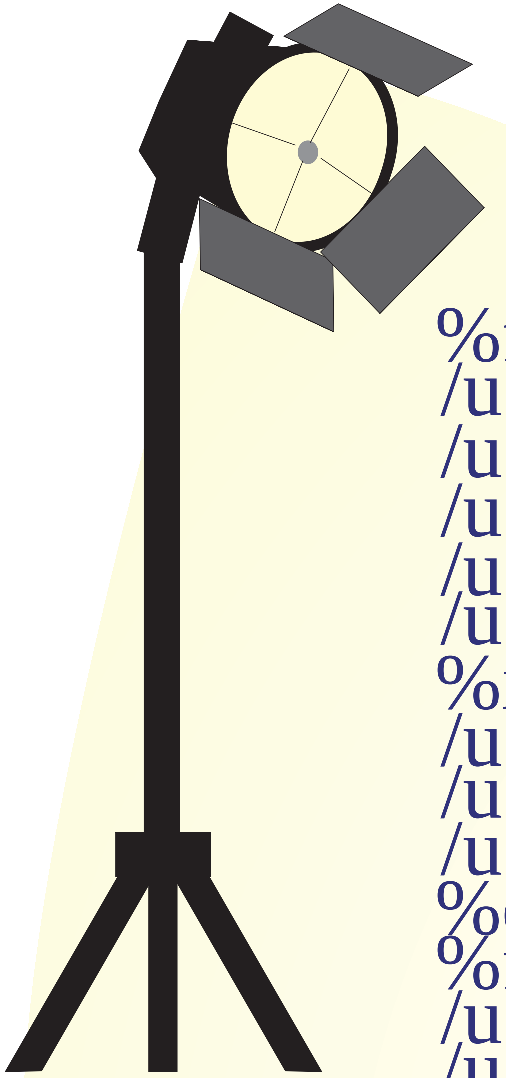
```
%install
rm -rf /usr/include/readline
%ifarch i386

make installall
ln -sf libreadline.so.2.0 /usr/lib/libreadline.so.2
ln -sf libreadline.so.2.0 /usr/lib/libreadline.so
ln -sf libhistory.so.2.0 /usr/lib/libhistory.so
%endif

%ifarch axp sparc
make install
%endif
```

Building GNU Readline: %install section (cont’d)

```
for n in history.info readline.info*; do
        cp doc/$n /usr/info
        gzip -nf9 /usr/info/$n
done
cp doc/readline.3 /usr/man/man3/readline.3
gzip -nf9 /usr/man/man3/readline.3
%ifarch i386
/sbin/ldconfig
%endif
```



Building GNU Readline: %files section

```
%files
/usr/man/man3/readline.3.gz
/usr/info/history.info.gz
/usr/info/readline.info.gz
/usr/info/readline.info-1.gz
/usr/info/readline.info-2.gz
%ifarch i386
/usr/lib/libreadline.so.2.0
/usr/lib/libreadline.so.2
/usr/lib/libhistory.so.2.0
%endif
%files devel
/usr/include/readline
/usr/lib/libreadline.a
%ifarch i386
/usr/lib/libreadline.so
/usr/lib/libhistory.so
%endif
```