5. Making packages

Thomas Lumley
Ken Rice

Universities of Washington and Auckland

Seattle, July 2015

Packages

The most important innovation in R over S-PLUS was the package system;

- Standard format for distributing code contributions, allows anyone to contribute.
- Cross-platform production of binaries
- Documentation format integrated into the R help system
- Increasing number of QA checks to improve code quality.
- Dependency tracking

Main documentation reference: *Writing R Extensions*

Package format

- `DESCRIPTION`, `NAMESPACE`: overall structure
- `R/` R code
- `src/` Code to be compiled
- `man/` help pages
- `vignettes/` Vignettes
- `data/` example data sets.
- `tests/` extra tests

R function `package.skeleton()` makes the basic structure for you.

DESCRIPTION

```
Package: survey
Title: analysis of complex survey samples
Description: Summary statistics, generalised linear models, cumulative link mode
ls, Cox models, loglinear models, and general maximum pseudolikelihood estimatio
n for multistage stratified, cluster-sampled, unequally weighted survey samples.
 Variances by Taylor series linearisation or replicate weights. Post-stratificat
ion, calibration, and raking. Two-phase subsampling designs. Graphics. Predictiv
e margins by direct standardization. PPS sampling without replacement. Principal
 components, factor analysis.
Version: 3.22-2
Author: Thomas Lumley
Maintainer: Thomas Lumley <tlumley@u.washington.edu>
License: GPL-2 | GPL-3
Depends: R (>= 2.2.0)
Suggests: survival, MASS, KernSmooth, hexbin, mitools, lattice, RSQLite, RODBC,
quantreg, splines, Matrix, multicore
URL: http://faculty.washington.edu/tlumley/survey/
```

DESCRIPTION

- `Package`: name of package
- `Title`: one-line description
- `Description`: one-paragraph description
- `Version`: X.YY-zz version number
- `Author`, `Maintainer`: the maintainer is the person to contact, needs email address
- `License`: preferably a standard abbreviation for a standard license if you are going to distribute the package.
- `Depends`: other packages needed to install this one, version of R needed
- `Suggests`: other packages this can make use of but doesn't need
- `URL`: web page for package

Also `Imports` for other packages imported in the `NAMESPACE`, `LazyLoad: yes` to allow loading-on-demand, `Encoding:` if the character set is not ASCII (eg `UTF-8`).

NAMESPACE

Controls which functions in the package are exported and visible to the user.

- Fewer name collisions and accidental redefinitions of a function
- Internal functions not intended for the user (and, eg, with less error checking) can be hidden
- S3 methods can be exported as methods rather than by name, so they can only be called via `UseMethod`
- Compiled code loaded in the NAMESPACE file will be found by `.C` and `.Call` only for functions in the package [session 7]

Example: `leaps` package

```
useDynLib(leaps)
export(regsubsets, leaps)
S3method(regsubsets, biglm)
S3method(regsubsets, formula)
S3method(regsubsets, default)
S3method(summary, regsubsets)
S3method(print, summary.regsubsets)
S3method(print, regsubsets)
S3method(plot, regsubsets)
S3method(coef, regsubsets)
S3method(vcov, regsubsets)
```

Example: `annotate` package [edited]

```
importClassesFrom(Biobase, eSet)

importMethodsFrom(AnnotationDbi,
                  Definition, GOID, Secondary,  Synonym, as.list, colnames, dbmeta
                  eapply,  exists, get,  ls, mappedRkeys, mget, ncol,  nrow,
                  Ontology, revmap, Term)
importMethodsFrom(Biobase,
                  annotation,  contents,  exprs, featureNames)

importFrom(Biobase,
           addVigs2WinMenu)
importFrom(graphics,
           abline, identify, plot)
importFrom(utils
           browseURL,  compareVersion, packageDescription)

exportClasses( chromLocation,  FramedHTMLPage,homoData,
               HTMLPage,  pubMedAbst )

exportMethods( abstText, articleTitle, authors, chromInfo, chromLengths,
               chromLocs, chromNames, dataSource, Definition,  fileName,
               geneSymbols, GOID,  homoACC,   homoHGID, homoLL,
         [snip]                 )
```

Example: `annotate` package [edited]

```
export(.buildAnnotateOpts, .getIdTag, .getNcbiURL, .handleXML,
       .pmfetch,   .transformAccession, ACC2homology, accessionToUID, ACCNUMStats,
       annPkgName, aqListGOIDs,  buildChromLocation,  buildPubMedAbst,
       checkArgs,  chrCats,  compatibleVersions,  createLLChrCats,
       createMAPIncMat,  dropECode, filterGOByOntology,  findChr4LL,
       findNeighbors,    genbank,    genelocator,  getAnnMap,
  [snip]
       setRepository, getRepositories, clearRepository, isValidKey, allValidKeys,
       updateSymbolsToValidKeys
       )
```

`import` makes available all functions exported from another package, but does not load that package (the functions are not visible to the user), `importFrom` imports just the specified functions.

`importClassesFrom`, `importMethodsFrom`, `exportClasses`, `exportMethods` are for S4 classes and methods

R/ subdirectory

R code, in files with extension `.r`, `.R`, `.q`, `.Q`, `.s` or `.S`

[Restriction on file names is to prevent, eg, editor backup files being accidentally used.]

Files are sorted in ASCII alphabetical order and concatenated into one file.

src/ subdirectory

R will try to compile

- C, Fortran 77: R uses these, so they work portably
- C++: Most systems with a C compiler also provide a compatible C++ compiler.
- Fortran 90, 95: Not all systems have Fortran 90/95 compilers. The linker needed for Fortran 95 is sometimes incompatible with C++ code.

Other types of code can be compiled by including a `Makefile`, but this tends not to be portable.

man/ subdirectory

Documentation for every user-visible object

- functions
- data objects
- S4 classes

Rd format

Similar to LaTeX

Markup language with tags distinguished by backslash: `\usage`, `\title`, `\arguments`

`prompt()` (or `package.skeleton()`) makes a skeleton help file for you to fill in, so you don't need to know that much of the format.

[show examples]

Vignettes

Help pages describe what a single function does, or the structure of a single class.

Vignettes describe how to do a task. They are longer than help pages and don't need to describe all the possibilities.

Vignettes are written in LaTeX or Markdown with specially formatted chunks of R. The `Sweave()` function (in the `tools` package, or the `knit()` function in the `knitr` package) extracts the R code chunks from the document, runs them, and then puts the output back into the document.

`latex` or `pdflatex` can then be used to make a PDF document.

Since the output in the document is automatically generated from the input, it is reliably correct, without cut-and-paste or version errors... a.k.a reproducible research!

data/ directory

This contains saved data sets that can be loaded with the `data()` function and used in help page examples.

Typically these are R binary data files created with the `save()` function and having a `.rda` or `.Rdata` extension, but R source files (`.R` extension) or white-space separated tables (`.txt` extension) are also allowed.

If the `DESCRIPTION` file has the line `LazyData: yes` the data files are automatically loaded when their names are used, and the `data()` function is not necessary.

Steps in making a package

1. You probably have at least partial code. Use `package.skeleton()` to set up a package directory with this code. Edit the help files. Put any code to be compiled in `src`

2. `R CMD INSTALL thepackage` installs the package

3. Run the functions and check that it works

4. `R CMD check thepackage` runs the QC tools

5. Fix all the errors and warnings, then go to step 2

6. `R CMD build thepackage` makes a package file for distribution to other people

R CMD check

Package QC checks

- Correct package structure, including portability of character sets.
- Documentation consistent with code
- Language rules not enforced by the parser, such as agreement in argument names between S3 methods and generics
- Absence of common coding errors
- All the help page examples run without errors
- Any tests in the `tests/` directory run correctly

Packages submitted to CRAN or Bioconductor should complete `R CMD check` in at most a minute or so: it may not be possible to put all your package tests in the `tests/` directory.

R-forge and winbuilder

http://win-builder.r-project.org/ will compile Windows binaries of your (working, checked) package: upload by ftp, it sends email when compilation is done.

http://r-forge.r-project.org/ hosts R package development:

- version control via `subversion`, including browseable code on web pages.
- mailing lists
- bug tracker
- daily build/check on Windows, Mac, 32-bit and 64-bit Linux