# Creating a package in R

## Using devtools and roxyen2

Brian Diggs

Portland R Users Group, August 22, 2012

# Evolution of wanting to create a package

- Write some code that solves a particular problem

# Evolution of wanting to create a package

- Write some code that solves a particular problem
- Run into same problem again on another project; cut and paste code between projects

# Evolution of wanting to create a package

- Write some code that solves a particular problem
- Run into same problem again on another project; cut and paste code between projects
- Same problem pops up a few more times; turn solution into function, generalize it somewhat, put the code in a separate file, `source` it

# Evolution of wanting to create a package

- Write some code that solves a particular problem
- Run into same problem again on another project; cut and paste code between projects
- Same problem pops up a few more times; turn solution into function, generalize it somewhat, put the code in a separate file, `source` it
- Colleague hears about your solution, wants to use it; send the file, hopefully with some comments added as to how it works

# Evolution of wanting to create a package

- Write some code that solves a particular problem
- Run into same problem again on another project; cut and paste code between projects
- Same problem pops up a few more times; turn solution into function, generalize it somewhat, put the code in a separate file, `source` it
- Colleague hears about your solution, wants to use it; send the file, hopefully with some comments added as to how it works
- Make some additional changes, maybe remember to update comments, forget to send updated version to colleague, think there must be a better way

# Evolution of wanting to create a package

- Write some code that solves a particular problem
- Run into same problem again on another project; cut and paste code between projects
- Same problem pops up a few more times; turn solution into function, generalize it somewhat, put the code in a separate file, `source` it
- Colleague hears about your solution, wants to use it; send the file, hopefully with some comments added as to how it works
- Make some additional changes, maybe remember to update comments, forget to send updated version to colleague, think there must be a better way

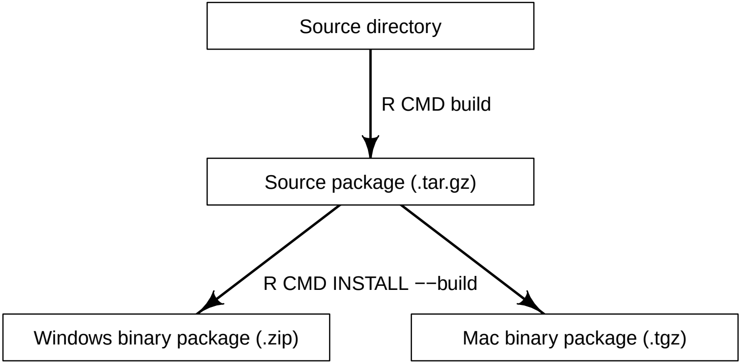
Package!

# Advantages of packages

- Source code organization
- Documentation
- Code isolation
- Testing infrastructure

# Aspects not covered

- Source code control/version control
- Compiled code in packages
- Submission to CRAN
- Internationalization
- Package maintenance and updating


Source directory
R CMD build
Source package (.tar.gz)
R CMD INSTALL −−build
Windows binary package (.zip)
Mac binary package (.tgz)

- Linux/Unix-like
  - http://cran.r-project.org/doc/manuals/R-admin.html#Essential-and-useful-other-programs-under-a-Unix_002dalike
  - Compilers
  - A LaTeX distribution
- Windows
  - http://cran.r-project.org/doc/manuals/R-admin.html#The-Windows-toolset
  - Rtools http://cran.r-project.org/bin/windows/Rtools/
  - MiKTeX, a LaTeX distribution http://www.miktex.org/
- Mac OS X
  - http://cran.r-project.org/bin/macosx/RMacOSX-FAQ.html#What-is-needed-to-build-R
  - Xcode from Apple
  - Fortran compiler http://cran.r-project.org/bin/macosx/tools/
  - MacTeX, a LaTeX distribution http://tug.org/mactex/

```
source("age.at.R", echo=TRUE, prompt="", continue="",
       max.deparse.length = Inf)

##
## age.at <- function(target = Sys.Date(), birth) {
##     targetlt <- as.POSIXlt(target)
##     birthlt <- as.POSIXlt(birth)
##     as.integer(targetlt$year - birthlt$year -
##                 1 * ((targetlt$mon < birthlt$mon) |
##                      ((targetlt$mon == birthlt$mon) &
##                       (targetlt$mday < birthlt$mday))))
## }
```

- Use `package.skeleton` to make the beginning of the package source
  - from files (`code_files` argument)
  - from objects in the environment (`list` argument, character vector of object names)
- This is done only once
- Thereafter, edit the files in the new source directory

```
package.skeleton(name = "age",
                 code_files = "age.at.R",
                 path = ".")

## Creating directories ...

## Creating DESCRIPTION ...

## Creating NAMESPACE ...

## Creating Read-and-delete-me ...

## Copying code files ...

## Making help files ...

## Done.

## Further steps are described in
## './age/Read-and-delete-me'.
```

```
list.files("age", recursive=TRUE)

## [1] "DESCRIPTION"        "NAMESPACE"
## [3] "R/age.at.R"         "Read-and-delete-me"
## [5] "man/age-package.Rd" "man/age.at.Rd"
```

- `DESCRIPTION` and `NAMESPACE` have specific formats
- Source code is under `R`
- Reference documentation is under `man`
- Will add and describe more directories as we go along

- Reference documentation
  - Required
  - Statically checked
  - `Rd` file format
  - in `man` directory
  - Accessible from within R by `help()`
- Tutorial or other freeform documentation (vignettes)
  - Optional
  - `Sweave` documents (`Rnw` source, PDF output)
  - in `vignette` directory; formerly in `inst/doc` directory
  - Accessible from within R by `vignette()`
- Other
  - Any format, PDF preferred
  - in `inst/doc` directory
  - Not directly accessible from within R

# roxygen2 and Rd files

Rd files document function by function

- Input parameters
- Output
- Short and long description

roxygen2 allows this same information to be documented in the same file as the function source code. The Rd file is then generated from this specification.

```
##' Compute the age, in years, on a given date given the birth date
##'
##' Age is an integer measure which increases each year on the month
##' and day corresponding to the month and day of birth. If the target
##' date is prior to the birthdate, a negative value is returned,
##' though this often indicates an error in the data. For birthdays on
##' leap day (February 29th), the age increase on March 1st.
##'
##' @param target a vector of dates in a format which is coercible to
##' a \code{\link{POSIXlt}} that represent the dates at which the age
##' is to be determined. Default is the current date.
##' @param birth a vector of dates in a format which is coercible to a
##' \code{\link{POSIXlt}} that are the corresponding birth dates.
##' @return an integer vector represening the corresponding ages at
##' the target time given the date of birth.
```

# Vignettes

- Vignettes are `Sweave` (`Rnw`) files
- Designed to dynamically use and display the results of the code in the package
- Supplemented with a % `VignetteIndexEntry{}` for indexing purposes

- Specifies what is meant to be public versus private
- Does not truly hide anything (can still be accessed)
- Can specify which functions from other packages this package relies on
- Also includes declaration of S3 generics and instances of S3 generics
- This can also be handled by `roxygen2` so as to keep it with the source code

```
##' @export
```

- Examples in documentation: part documentation, part testing
    - Are run (by default) as part of package checking
    - Only tests that an error is not thrown
    - Can be run in an interactive session with `example()` to see output
- Explicit tests
    - In the `tests` directory
    - Can hook into testing suites such as the `testthat` package
    - Can be used to verify results of the examples
- Vignettes act as a sort of test in that failure to build is not allowed

# Examples

```
##' @examples
##' birthdates <- as.Date(c("1915-06-16", "1919-10-18", "1890-02-17", "
##' targetdates <- as.Date(c("1990-03-05", "1992-05-07", "1994-07-09",
##' age.at(targetdates, birthdates)
##' age.at(as.Date("1990-03-05"), birthdates)
##' age.at(targetdates, as.Date("1919-10-18"))
##' age.at(birth=birthdates)
```

## tests/test-all.R

```
library("testthat")
library("age")

test_package("age")
```

# inst/tests/test-age-at.R

```
context("age.at")

test_that("increment on birthday", {
    expect_that(age.at(as.Date("2010-10-15")+(-5:5),
                       as.Date("1960-10-15")),
                equals(c(rep(49L,5), rep(50L,6))))
    expect_that(age.at(as.Date("2010-10-01")+(-5:5),
                       as.Date("1960-10-01")),
                equals(c(rep(49L,5), rep(50L,6))))
    expect_that(age.at(as.Date("2010-10-31")+(-5:5),
                       as.Date("1960-10-31")),
                equals(c(rep(49L,5), rep(50L,6))))
})

test_that("leap day birthday", {
    expect_that(age.at(as.Date("2010-03-01")+(-5:5),
                       as.Date("1960-02-29")),
                equals(c(rep(49L,5), rep(50L,6))))
})
```

Doing all these things, our package directory now looks like

```
list.files("age", recursive=TRUE)

## [1] "DESCRIPTION"
## [2] "NAMESPACE"
## [3] "R/age.at.R"
## [4] "inst/tests/test-age-at.R"
## [5] "man/age-package.Rd"
## [6] "man/age.at.Rd"
## [7] "tests/test-all.R"
```

# Turning the package source into a source package

```
document("age")

## Updating age documentation

## Updating collate directive in  /Users/schnade/Desktop/Create-R-packa
## Updating namespace directives
## Writing age.at.Rd

build("age")

## [1] "/Users/schnade/Desktop/Create-R-package/age_1.0.tar.gz"

file.exists("age_1.0.tar.gz")

## [1] TRUE
```

# Checking the package

```
check("age")

## Updating age documentation

## Writing age.at.Rd

## Checking age
```

- *Writing R Extensions*
  `http://cran.r-project.org/doc/manuals/R-exts.html`
- Hadley Wickham's `devtools` wiki
  `https://github.com/hadley/devtools/wiki`

# Session info

```r
toLatex(sessionInfo())
```

- R version 2.15.1 (2012-06-22), `x86_64-apple-darwin9.8.0`
- Locale: `C`
- Base packages: base, datasets, grDevices, graphics, methods, stats, tools, utils
- Other packages: devtools 0.7.1, diagram 1.6, digest 0.5.2, knitr 0.7, roxygen2 2.2.2, shape 1.4.0
- Loaded via a namespace (and not attached): RCurl 1.91-1, brew 1.0-6, evaluate 0.4.2, formatR 0.6, httr 0.1.1, memoise 0.1, plyr 1.7.1, stringr 0.6.1