

Lec 26 - R Packages

Statistical Programming

Sta 323 | Spring 2022

Dr. Colin Rundel

What are R packages?

R packages are just a collection of files (R code, compiled code, data, documentation, etc.) that live in your library path.

```
.libPaths()
```

```
## [1] "/opt/homebrew/lib/R/4.1/site-library"      "/opt/homebrew/Cellar/r/4.1.3/lib/R/library"
```

When you run `library(pkg)` the functions (and objects) in the package's namespace are attached to the global search path.

```
dir(.libPaths())
```

```
## [1] "abind"           "airports"        "archive"         "arrayhelpers"    "arrow"
## [6] "ash"            "AsioHeaders"     "askpass"         "assertthat"      "av"
## [11] "available"      "babynames"       "backports"       "base"            "base64enc"
## [16] "BayesFactor"    "bayesplot"       "beeswarm"        "bench"           "benchmarkme"
## [21] "benchmarkmeData" "BH"              "bit"             "bit64"           "bitops"
## [26] "blob"          "bookdown"        "boot"            "boot"            "brew"
## [31] "brio"          "broom"           "broom.mixed"     "bslib"           "cachem"
## [36] "callr"         "car"             "carData"         "caret"           "caTools"
## [41] "cellranger"    "checklist"       "checkmate"       "cherryblossom"   "chromote"
## [46] "chron"         "class"           "class"           "classInt"        "cli"
```

Search path

```
search()
```

```
## [1] ".GlobalEnv"      "package:stats"    "package:graphics" "package:grDevices" "package:utils"  
## [6] "package:datasets" "package:methods"  "Autoloads"        "package:base"
```

```
library(diffmatchpatch)
```

```
search()
```

```
## [1] ".GlobalEnv"      "package:diffmatchpatch" "package:stats"      "package:graphics"  
## [5] "package:grDevices" "package:utils"          "package:datasets"   "package:methods"  
## [9] "Autoloads"        "package:base"
```

Loading vs attaching

If you do not want to attach a package you can directly use functions via `::` or load it with `requireNamespace()`.

```
loadedNamespaces()
```

```
## [1] "Rcpp"          "grDevices"    "digest"       "diffmatchpatch" "R6"           "jsonlite"
## [7] "magrittr"     "evaluate"     "datasets"     "xaringan"      "stringi"      "rlang"
## [13] "utils"        "cli"          "rstudioapi"   "jquerylib"     "bslib"        "graphics"
## [19] "rmarkdown"    "base"         "tools"        "stringr"       "xfun"         "yaml"
## [25] "fastmap"      "compiler"     "stats"        "htmltools"     "knitr"        "methods"
## [31] "sass"
```

```
requireNamespace("forcats")
```

```
## Loading required namespace: forcats
```

```
loadedNamespaces()
```

```
## [1] "Rcpp"          "grDevices"    "digest"       "diffmatchpatch" "R6"           "jsonlite"
## [7] "magrittr"     "evaluate"     "datasets"     "xaringan"      "stringi"      "rlang"
## [13] "utils"        "cli"          "rstudioapi"   "jquerylib"     "bslib"        "graphics"
## [19] "rmarkdown"    "base"         "forcats"      "tools"         "stringr"      "xfun"
## [25] "yaml"         "fastmap"      "compiler"     "stats"         "htmltools"    "knitr"
## [31] "methods"     "sass"
```

```
search()
```

```
## [1] ".GlobalEnv"      "package:diffmatchpatch" "package:stats"      "package:graphics"
## [5] "package:grDevices" "package:utils"          "package:datasets"   "package:methods"
```

Where to R packages come from

We've already seen the two primary sources of R packages:

CRAN:

```
install.packages("diffmatchpatch")
```

GitHub:

```
remotes::install_github("rundle/diffmatchpatch")
```

there is one other method that comes up (particularly around package development), which is to install a package from local files.

Local install:

```
R CMD install diffmatchpatch_0.1.0.tar.gz
```

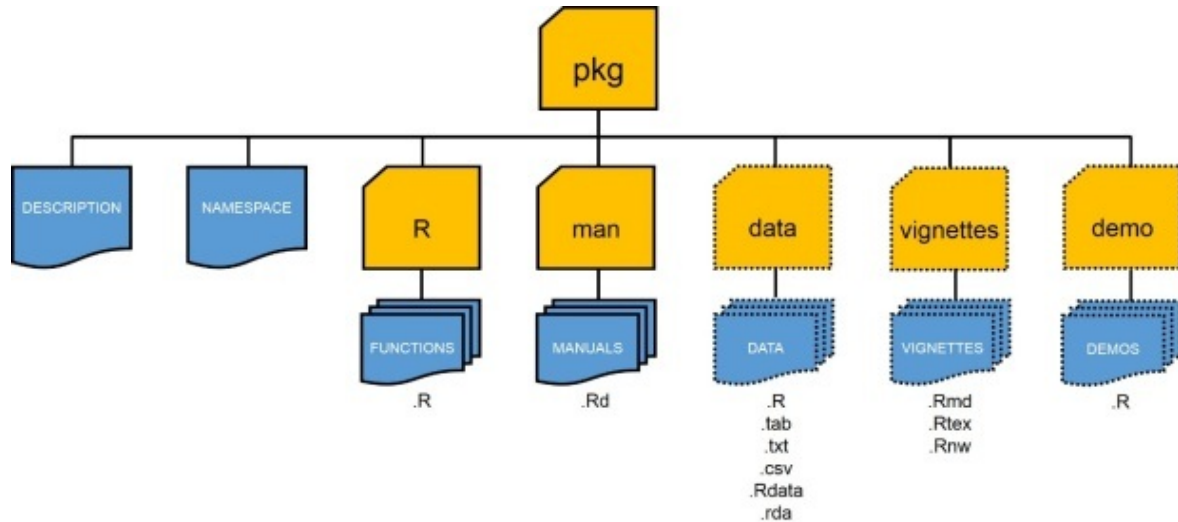
```
devtools::install("diffmatchpatch_0.1.0.tar.gz")
```

What is CRAN

It is the Comprehensive R Archive Network which is the central repository of R packages.

- Maintained by the R Foundation and run by a team of volunteers, ~22k packages
- Retains all current versions of released packages as well as archives of previous versions
- Similar in spirit to Perl's CPAN, TeX's CTAN, and Python's PyPI
- Some important features:
 - All submissions are reviewed by humans + automated checks
 - Strictly enforced submission policies and package requirements
 - All packages must be actively maintained and support upstream and downstream changes

Structure of an R Package



Core components

- DESCRIPTION - file containing package metadata (e.g. package name, description, version, license, and author details). Also specifies package dependencies,
- NAMESPACE - details which functions and objects are exported by your package
- R/ - folder containing R script files (.R)
- man/ - folder containing R documentation files (.Rd)

The following components are optional, but quite common:

- tests/ - folder contain unit tests
- src/ - folder containing code to be compiled (usually C / C++)
- data/ - folder containing example data sets (exported as .Rdata via `save()`)
- inst/ - files that will be copied to the package's top-level directory when it is installed (e.g. examples or data files that don't belong in data/)

Package contents

Source Package

```
fs::dir_tree("~/Desktop/Projects/diffmatchpatch/")
```

```
## ~/Desktop/Projects/diffmatchpatch/
## └── DESCRIPTION
## └── LICENSE.md
## └── NAMESPACE
## └── NEWS.md
## └── R
##     ├── RcppExports.R
##     ├── diff.R
##     ├── diffmatchpatch-package.R
##     ├── match.R
##     ├── options.R
##     ├── patch.R
##     └── print.R
## └── README.Rmd
## └── README.md
## └── cran-comments.md
## └── diffmatchpatch.Rproj
## └── inst
##     └── include
##         └── diff_match_patch.h
## └── man
##     ├── diff.Rd
##     ├── dmp_options.Rd
##     ├── match.Rd
##     └── patch.Rd
```

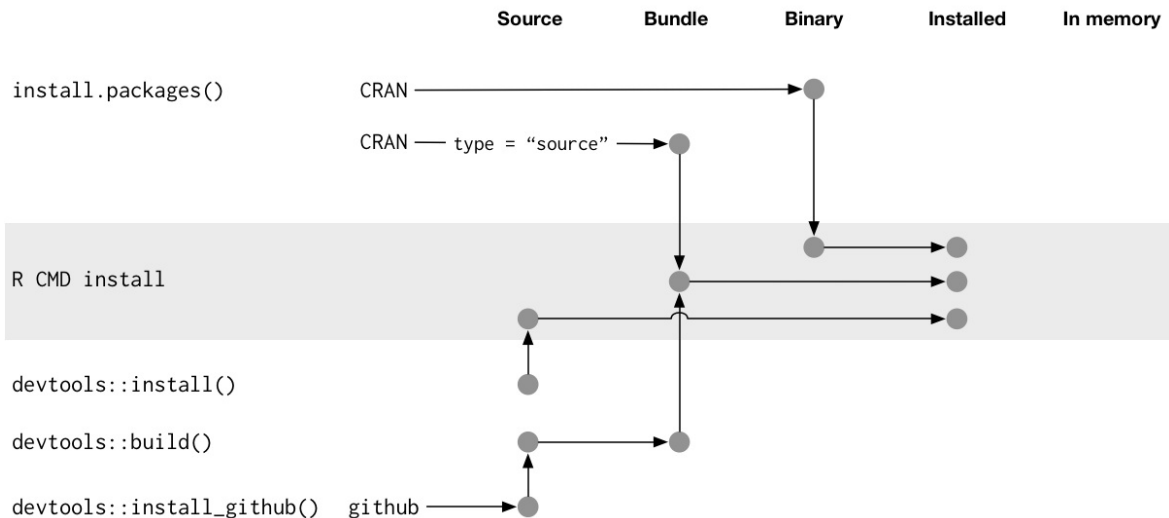
Installed Package

```
fs::dir_tree(system.file(package="diffmatchpatch"))
```

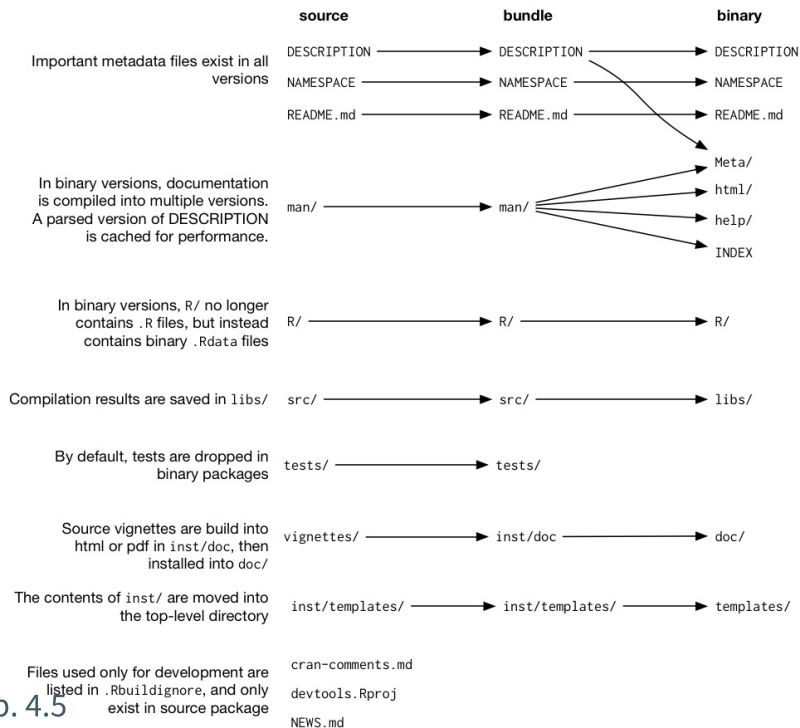
```
## /opt/homebrew/lib/R/4.1/site-library/diffmatchpatch
## └── DESCRIPTION
## └── INDEX
## └── Meta
##     ├── Rd.rds
##     ├── features.rds
##     ├── hsearch.rds
##     ├── links.rds
##     ├── nsInfo.rds
##     └── package.rds
## └── NAMESPACE
## └── NEWS.md
## └── R
##     ├── diffmatchpatch
##     ├── diffmatchpatch.rdb
##     └── diffmatchpatch.rdx
## └── help
##     ├── AnIndex
##     ├── aliases.rds
##     ├── diffmatchpatch.rdb
##     ├── diffmatchpatch.rdx
##     └── paths.rds
## └── html
##     ├── 00Index.html
##     └── R.css
```

A deeper dive on diffmatchpatch

Package Installation



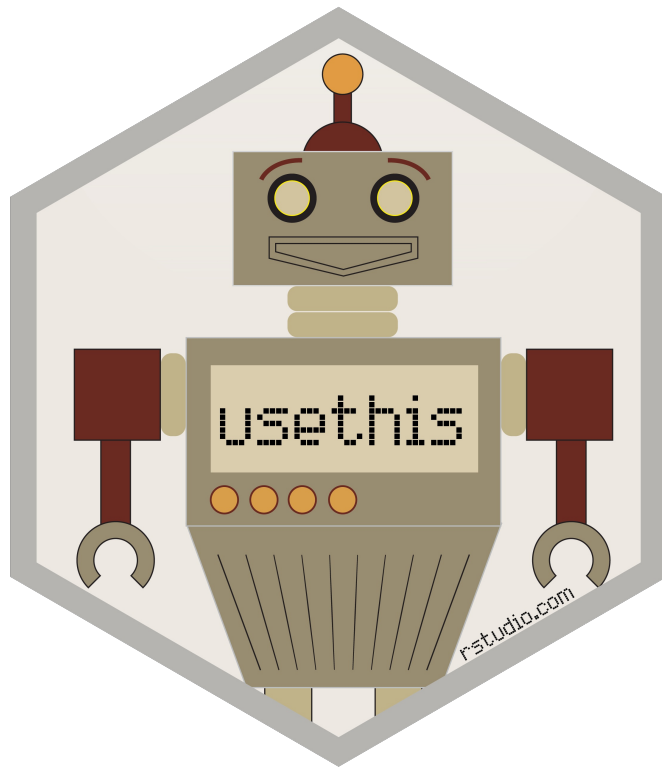
Package Installation - Files



Package development

What follows is an opinionated introduction to package development,

- this is not the only way to do thing (none of the following are required)
- I would strongly recommend using:
 - RStudio
 - RStudio projects
 - GitHub
 - usethis
 - roxygen2



usethis

This is an immensely useful package for automating all kinds of routine (and tedious) tasks within R

- Tools for managing git and GitHub configuration
- Tools for managing collaboration on GitHub via pull requests (see `pr_*`())
- Tools for creating and configuring packages
- Tools for configuring your R environment (e.g. `.Rprofile` and `.Renv`)
- and much much more

Live demo - Building a Package