# An Introduction to Developing R Packages

Christina P. Knudson[1] (knud8583@stthomas.edu), Lindsey R. Dietz[2] (lindsey.dietz@mpls.frb.org) and Haema Nilakanta[3] (nilak008@umn.edu)
[1]Department of Mathematics, University of St. Thomas; [2]Federal Reserve Bank of Minneapolis; [3]School of Statistics, University of Minnesota

## Introduction

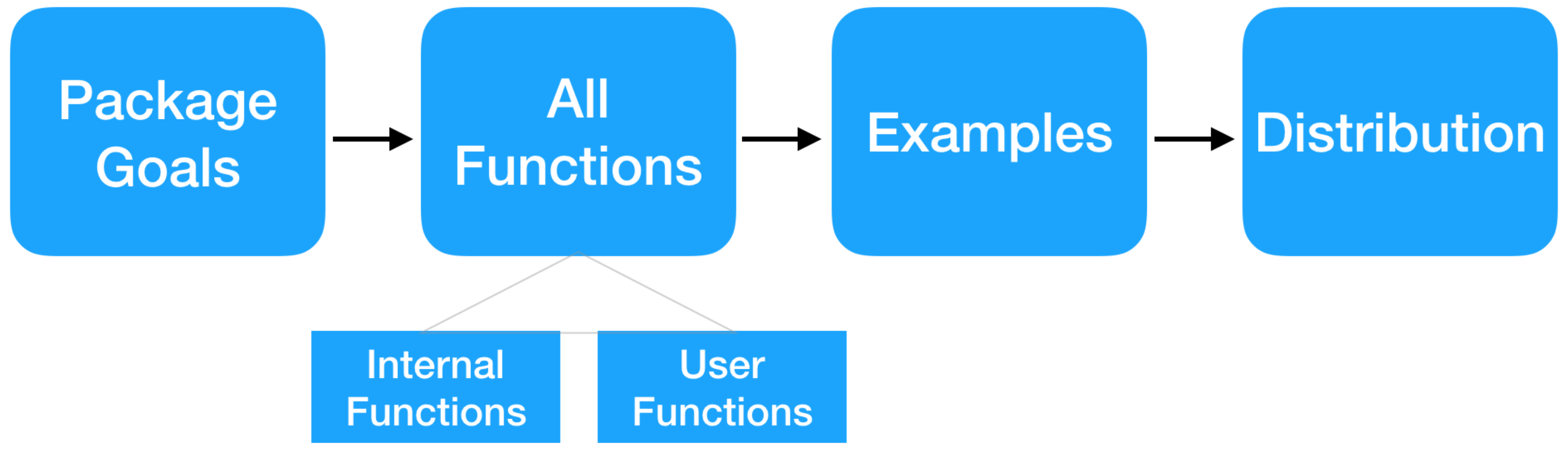
Only about ten percent of R packages are maintained by women: this poster aims to turn this trend around. We will briefly cover the steps of planning, developing and maintaining R packages. We will guide you through these steps via a basic package we created called `wisdom`. Fellow researchers can download this package from a public GitHub page and use it as a template for their future packages.

## Planning

Why make an R package?

- Efficiency - "package" your functions
- Reproducibility - sound scientific research
- Portability - share your work

Start by making a *design document* that consists of a plan that includes:



For our package:

- **Package Name:** `wisdom`
- **Package Goals:** Create a simple easy-to-use package for WSDS 2017 that fellow researchers can use as a template
- **Functions (user):** `mpower(x,m)` raises a base value $x$ to the power $m$ and outputs the result

## Writing Functions

After finishing the design document, write all the functions involved (both internal and user). `wisdom` has only one function:

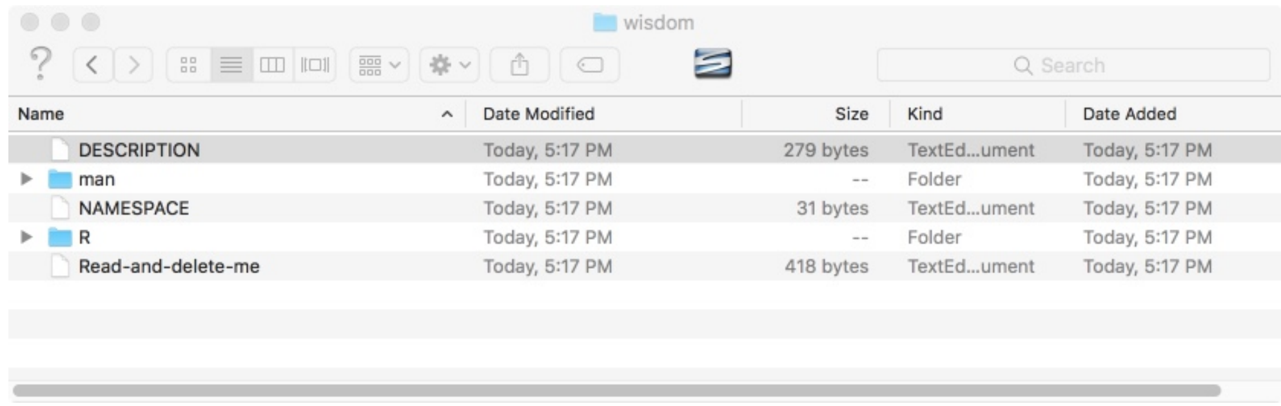
```
mPower <- function(x,m){
    x^m
}
```

## Skeleton

After writing the function(s), create a *skeleton* for the package (with package name and list of functions to include).

```
> package.skeleton(name="wisdom", list='mPower')
Creating directories ...
Creating DESCRIPTION ...
Creating NAMESPACE ...
Creating Read-and-delete-me ...
Saving functions and data ...
Making help files ...
Done.
Further steps are described in './wisdom/Read-and-delete-me'.
```

This automatically generates all the files for the package you will need.



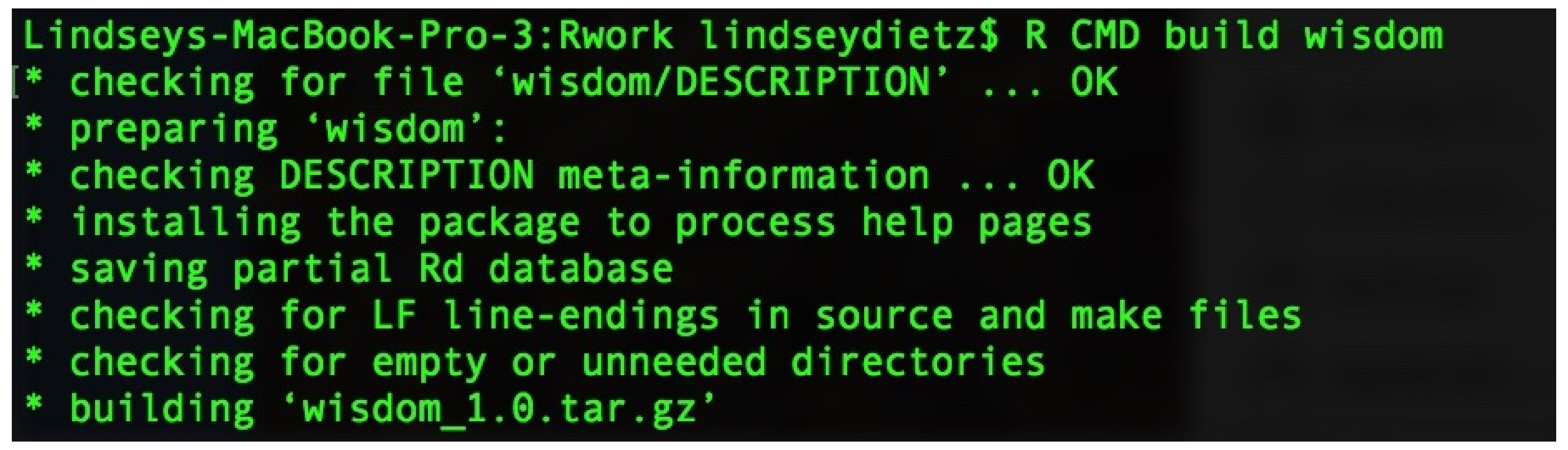
| Name | Date Modified | Size | Kind | Date Added |
|---|---|---|---|---|
| DESCRIPTION | Today, 5:17 PM | 279 bytes | TextEd...ument | Today, 5:17 PM |
| man | Today, 5:17 PM | -- | Folder | Today, 5:17 PM |
| NAMESPACE | Today, 5:17 PM | 31 bytes | TextEd...ument | Today, 5:17 PM |
| R | Today, 5:17 PM | -- | Folder | Today, 5:17 PM |
| Read-and-delete-me | Today, 5:17 PM | 418 bytes | TextEd...ument | Today, 5:17 PM |

- `Description`: package description file
- `man` folder: folder of all user function manuals
- `NAMESPACE`: lists all functions available to the user
- `R` folder: contains all functions (internal and user)

## Building & Checking

When you have completed all package code and documentation, you can build and check your package.

**Build:** create a *tarball* file, i.e., a zip of all the items associated with the package. In the terminal window, navigate to the directory where package files are located (with the `cd` command) and type: `R CMD build wisdom`



```
Lindseys-MacBook-Pro-3:Rwork lindseydietz$ R CMD build wisdom
* checking for file ‘wisdom/DESCRIPTION’ ... OK
* preparing ‘wisdom’:
* checking DESCRIPTION meta-information ... OK
* installing the package to process help pages
* saving partial Rd database
* checking for LF line-endings in source and make files
* checking for empty or unneeded directories
* building ‘wisdom_1.0.tar.gz’
```

**Check:** to ensure it built correctly, in the same terminal window type: `R CMD check wisdom_1.0.tar.gz`

Lots of lines will print out; do not fret! A line ending with `OK` means no errors; otherwise check the log file for issues.

## Distribution

After completing the build and check steps, there are two main ways to distribute the R package:

1. **CRAN:** requires submission to CRAN with extra checks and proper documentation.In the terminal window where the *tarball* is located type:
   `R CMD check wisdom_1.0.tar.gz -as-cran`
   - Pro: users can easily install packages from R and packages are viewed as more reliable since they go through several checks
   - Con: more regulated so takes longer to distribute and more maintenance required
2. **GitHub:** can host on public/personal webpage
   - Pro: quick distribution and authors can maintain development branches R
   - Con: indirect installation into R

## Other Items

Items to consider not gone into detail here:

- Including test functions
- Improving computational stability
- Choosing a license
- Writing vignettes to guide users
- Updating the package

## Resources

Download `widsom`, a package to `mPower` women in statistics and data science!

Other resources:

- Poster handout with more details
- Hadley Wickham's free online textbook: *R packages*
- R Extensions website