Easy package development

Hadley Wickham

Assistant Professor / Dobelman Family Junior Chair

Department of Statistics / Rice University

June 2012

HELLO

my name is

Hadley

Winston Chang
Research Associate
Department of Statistics
Rice University

Caveats

- Opinionated advice – not everyone agrees that this is the right way to build packages.
- Don’t look at ggplot2 (yet)
- This class only scratches the surface – many more details to learn.

Outline

- Package basics: `devtools`
- Documentation: `roxygen2`
- Testing: `testthat`
- Releasing your package: `devtools`

Getting started

```
# Check that you're ready to go

library(devtools)
has_devel()

# If not:
#   * on windows: install Rtools
#   * on mac: install xcode
#   * on linux: install gcc, make

library(testthat)
library(roxygen2)
```

Packages to look at

- `devtools`, `evaluate`, `lubridate`, `reshape2`, `stringr`, `testthat`: source code for my packages
- `coin`, `Matrix`: illustrate other important parts of packages that I don't use
- `one-file`, `minimal`, `...`: various stages of a package you'll be developing today

Learn from others

If you only remember one thing:

Read the source of other packages!

`https://github.com/hadley/plyr`

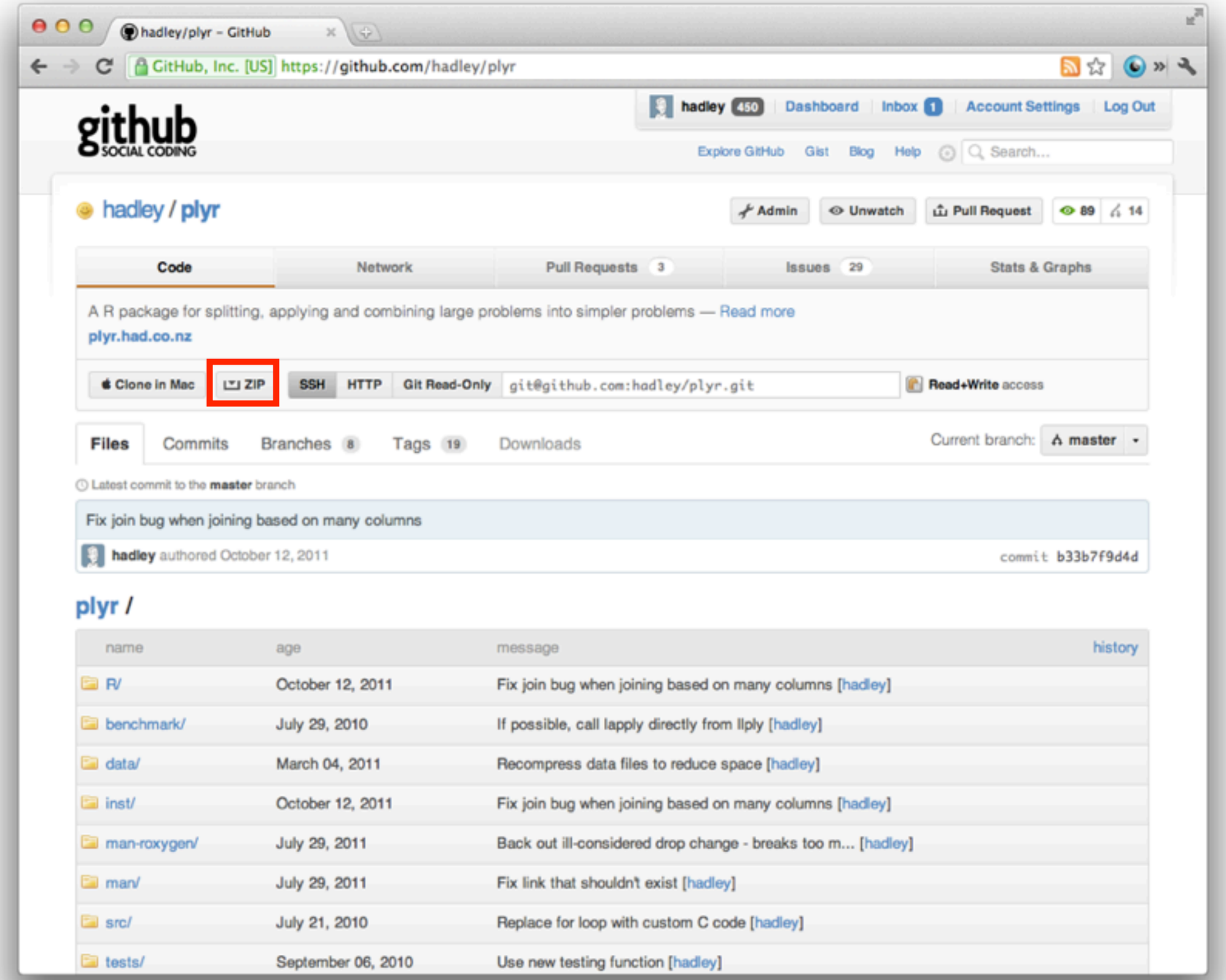
`https://github.com/hadley/stringr`

`https://github.com/hadley/devtools`

`https://github.com/hadley/lubridate`

`https://github.com/hadley/evaluate`

`https://github.com/hadley/reshape`


hadley/plyr - GitHub
GitHub, Inc. [US] https://github.com/hadley/plyr
github
SOCIAL CODING
hadley 450 Dashboard Inbox 1 Account Settings Log Out
Explore GitHub Gist Blog Help Search...
hadley / plyr
Admin Unwatch Pull Request 89 14
Code Network Pull Requests 3 Issues 29 Stats & Graphs
A R package for splitting, applying and combining large problems into simpler problems — Read more
plyr.had.co.nz
Clone in Mac ZIP SSH HTTP Git Read-Only git@github.com:hadley/plyr.git Read+Write access
Files Commits Branches 8 Tags 19 Downloads Current branch: master
Latest commit to the master branch
Fix join bug when joining based on many columns
hadley authored October 12, 2011 commit b33b7f9d4d
plyr /
name age message history
R/ October 12, 2011 Fix join bug when joining based on many columns [hadley]
benchmark/ July 29, 2010 If possible, call lapply directly from llply [hadley]
data/ March 04, 2011 Recompress data files to reduce space [hadley]
inst/ October 12, 2011 Fix join bug when joining based on many columns [hadley]
man-roxygen/ July 29, 2011 Back out ill-considered drop change - breaks too m... [hadley]
man/ July 29, 2011 Fix link that shouldn't exist [hadley]
src/ July 21, 2010 Replace for loop with custom C code [hadley]
tests/ September 06, 2010 Use new testing function [hadley]

CRAN - Package plyr

cran.r-project.org/web/packages/plyr/index.html

plyr: Tools for splitting, applying and combining data

plyr is a set of tools that solves a common set of problems: you need to break a big problem down into manageable pieces, operate on each pieces and then put all the pieces back together. For example, you might want to fit a model to each spatial location or time point in your study, summarise data by panels or collapse high-dimensional arrays to simpler summary statistics. The development of plyr has been generously supported by BD (Becton Dickinson).

Version:	1.6
Depends:	R (≥ 2.11.0)
Imports:	itertools, iterators
Suggests:	abind, testthat (≥ 0.2), tcltk, foreach
Published:	2011-07-29
Author:	Hadley Wickham
Maintainer:	Hadley Wickham <h.wickham at gmail.com>
License:	MIT
URL:	http://had.co.nz/plyr
Citation:	plyr citation info
CRAN checks:	plyr results

Downloads:

Package source:	plyr_1.6.tar.gz
MacOS X binary:	plyr_1.6.tgz
Windows binary:	plyr_1.6.zip
Reference manual:	plyr.pdf
News/ChangeLog:	NEWS
Old sources:	plyr archive

Your turn

Download and extract the source code for one R package that you use regularly. Read the source code for the function from that package that use most regularly. What did you learn?

Working directory

Why?

All paths in R are relative to the working directory. Life is much easier when you have it correctly set.

Usually want one project per directory. (See also Rstudio's project support)

Makes code easy to move between computers. **Never** use `setwd()` in a script.

How?

Rstudio: Tools | Set working directory | Choose directory ... (^⇧K)

Windows: File | Change dir. For frequent use, make shortcut in that folder.

Mac: ⌘D

Terminal: start R from the desired directory

```
# Find out what directory you’re in
getwd()
```

```
# List files in that directory
dir()
```

Your turn

Make sure your working directory is set to the `rv` directory inside the code and data directory you downloaded. Use `dir()` to check you're in the right place.

Projects

- RStudio's projects make this even easier: double click on Rproj file to set working directory and restore state
- Also enables project search

Discrete random variables

Definitions

A **random variable** is a random experiment with a numeric sample space. Usually given a capital letter like X, Y or Z.

(More formally a random variable is a function that converts outcomes from a random experiment into numbers)

The **space** (or **support**) of a random variable is the range of the function (cf. sample space)

$$P(a < X < b) = \sum_{x_i \in (a,b)} f(x_i)$$

$$P(X = x) = f(x)$$

probability pmf

To be a pmf, f must satisfy two conditions:

$$\sum_{x_i \in S} f(x_i) = 1$$

$$f(x_i) \geq 0, \forall\ x_i \in S$$

Transformations

x	-1	0	1	2	3
f(x)	0.2	0.1	0.3	0.1	0.3

Let X be a discrete random variable with pmf f as defined above.

Write out the pmfs for:

$A = X + 2 \quad B = 3X \quad C = X^2 \quad D = 0 * X$

Mean & variance

The **mean** summarises the “middle” of the distribution. The **variance** summarises the “spread” of the distribution.

Mean = E(X) = “Sum” of all outcomes, weighted by their probability.

Variance = Var(X) = E[$(X - E[X])^2$)] = expected squared distance from mean

Expectation

Expectation is a **linear operator**:

Expectation of a sum =
sum of expectations (additive)

Expectation of a constant * a function =
constant * expectation of function (homogenous)

Expectation of a constant is a constant.