

Understanding RStudio Add-ins

Jonathan McPherson

March 9, 2016

About me

- jonathan@rstudio.com
- RStudio IDE engineer for about 3 years

Goals

- Powerful, native extension mechanism
- Build reproducible tools over formerly ad-hoc, manual workflows
- Keep the core experience clean
- Easy to build, easy to share

What's an add-in?

- New in RStudio 0.99.891
- Written in R
- Lives in an R package
- Always invoked by the user (no background or event hooks)
- Can interact with the user's workspace through API
- Can be interactive or non-interactive

What can you build?

- Code builder tools – build up statements interactively
- Editor tools – formatting, templates, etc.
- Integrate with 3rd party data or code tools
- Kick off scripts or external jobs
- Integration with your company's systems
- Pretty much anything

Examples

- [Copy data frames to the clipboard](#)
- [Render R Markdown documents at the console](#)
- [Wrap R Markdown text without breaking inline R](#)
- [Assign function defaults for debugging](#)
- [Beautify R code](#)
- [Interactive data manipulation with TidyR](#)
- [Interactive in-document refactor](#)
- [Auto-generate Roxygen skeleton for functions & data](#)

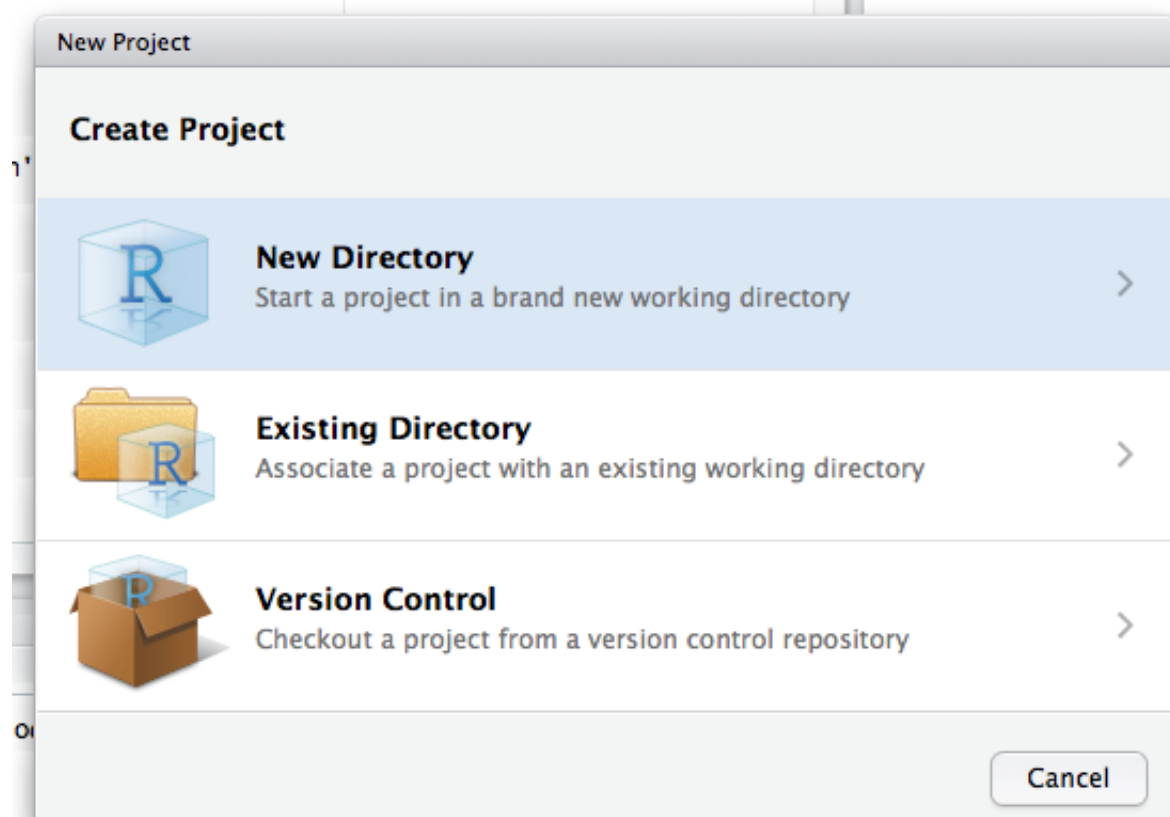
Anatomy of an add-in

Just 3 pieces:

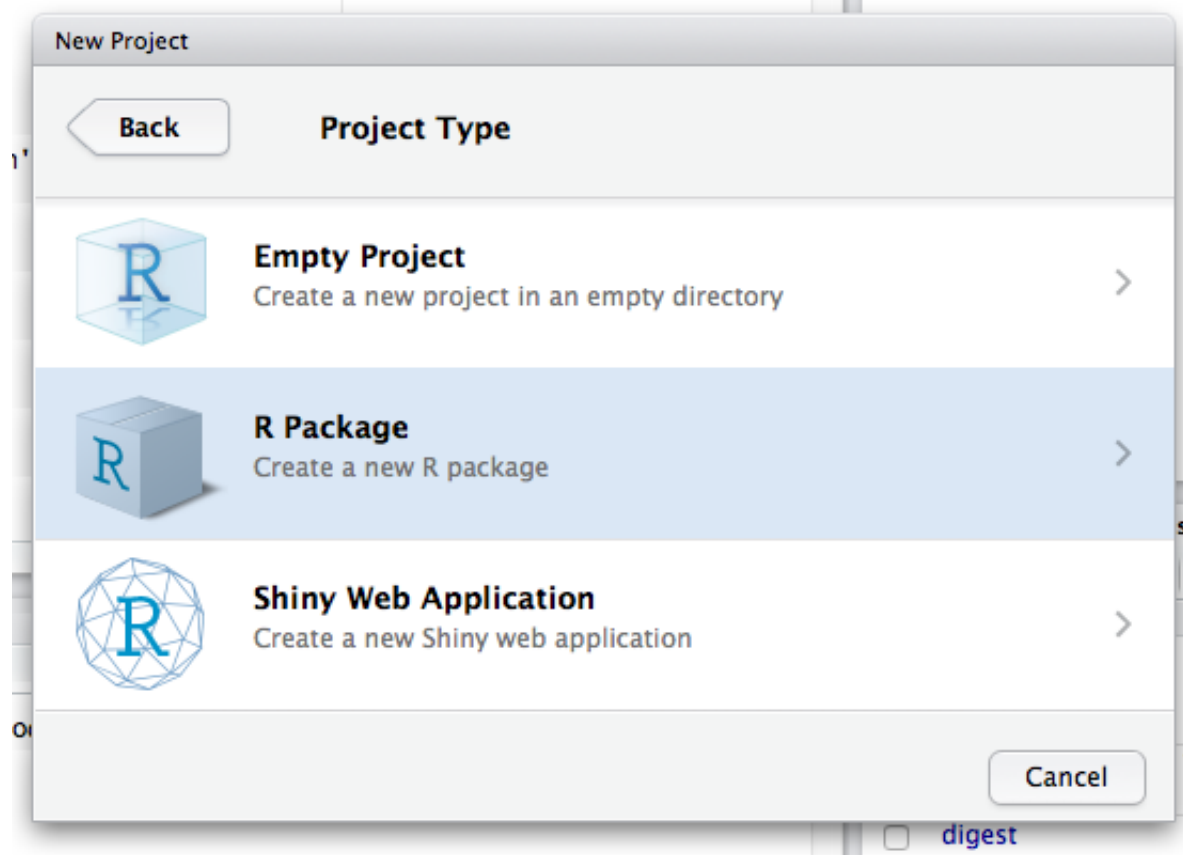
1. An R package (new or existing)
2. A `function()` that does the add-in work, invoking functions from the `rstudioapi` package to interact with the IDE.
3. A DCF named `inst/rstudio/addins.dcf` that declares the add-in's metadata

Let's build one to increment and decrement selected numbers.

Create the project




Make it a package



Give it a name

New Project

[Back](#) **Create R Package**



Type: Package Package name:

Create package based on source files:

[Add...](#)
[Remove](#)

Create project as subdirectory of:

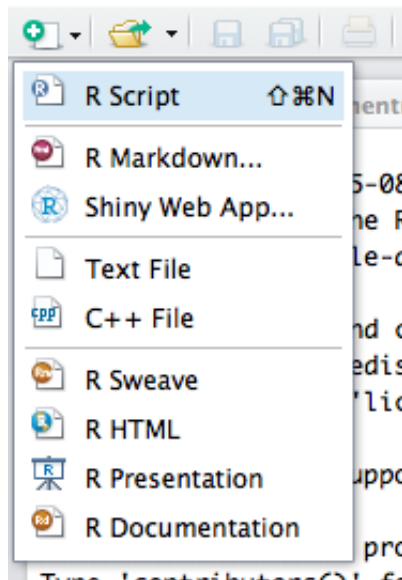
[Browse...](#)

☒ Create a git repository ☐ Use packrat with this project

☐ Open in new session

[Create Project](#) [Cancel](#)

Add an R script to the project



Add an empty function

```
increment <- function(delta) {  
}
```

Get the active document context

```
increment <- function(delta) {  
  context <- rstudioapi::getActiveDocumentContext()  
}
```

Loop over all the selections

```
increment <- function(delta) {  
  context <- rstudioapi::getActiveDocumentContext()  
  for (sel in context$selection) {  
  
  }  
}
```

Extract the integer from the selection

```
increment <- function(delta) {  
  context <- rstudioapi::getActiveDocumentContext()  
  for (sel in context$selection) {  
  
    suppressWarnings(int <- as.integer(sel$text))  
    if (is.na(int)) next  
  
  }  
}
```

Write the integer back into the document

```
increment <- function(delta) {  
  context <- rstudioapi::getActiveDocumentContext()  
  for (sel in context$selection) {  
  
    suppressWarnings(int <- as.integer(sel$text))  
    if (is.na(int)) next  
  
    rstudioapi::modifyRange(sel$range,  
                           as.character(int + delta),  
                           context$id)  
  
    break  
  }  
}
```


Add functions we can call from the add-in

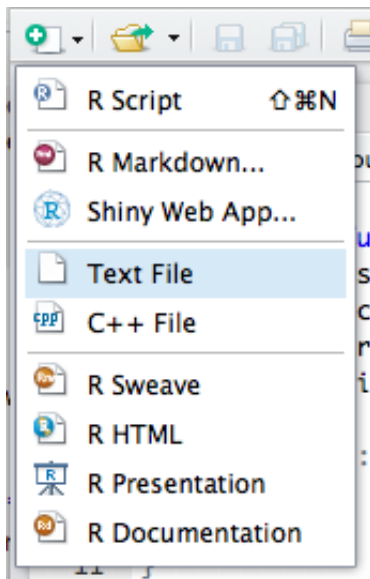
```
incrementr <- function() {  
  increment(1)  
}
```

```
decrementr <- function() {  
  increment(-1)  
}
```

Create the add-ins installation folder

```
> dir.create("inst/rstudio", recursive = TRUE)
```

Create a new text file and save as **addins.dcf**



Add entry for increment function

Name: Increment Integer

Description: Increments the selected number.

Binding: incrementr

Interactive: false

Add entry for decrement function

Name: Increment Integer

Description: Increments the selected number.

Binding: incrementr

Interactive: false

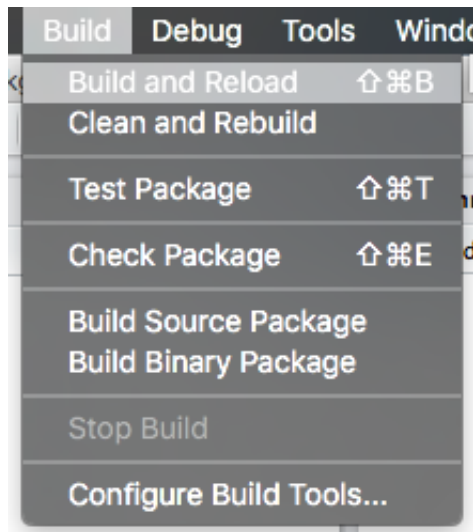
Name: Decrement Integer

Description: Decrements the selected number.

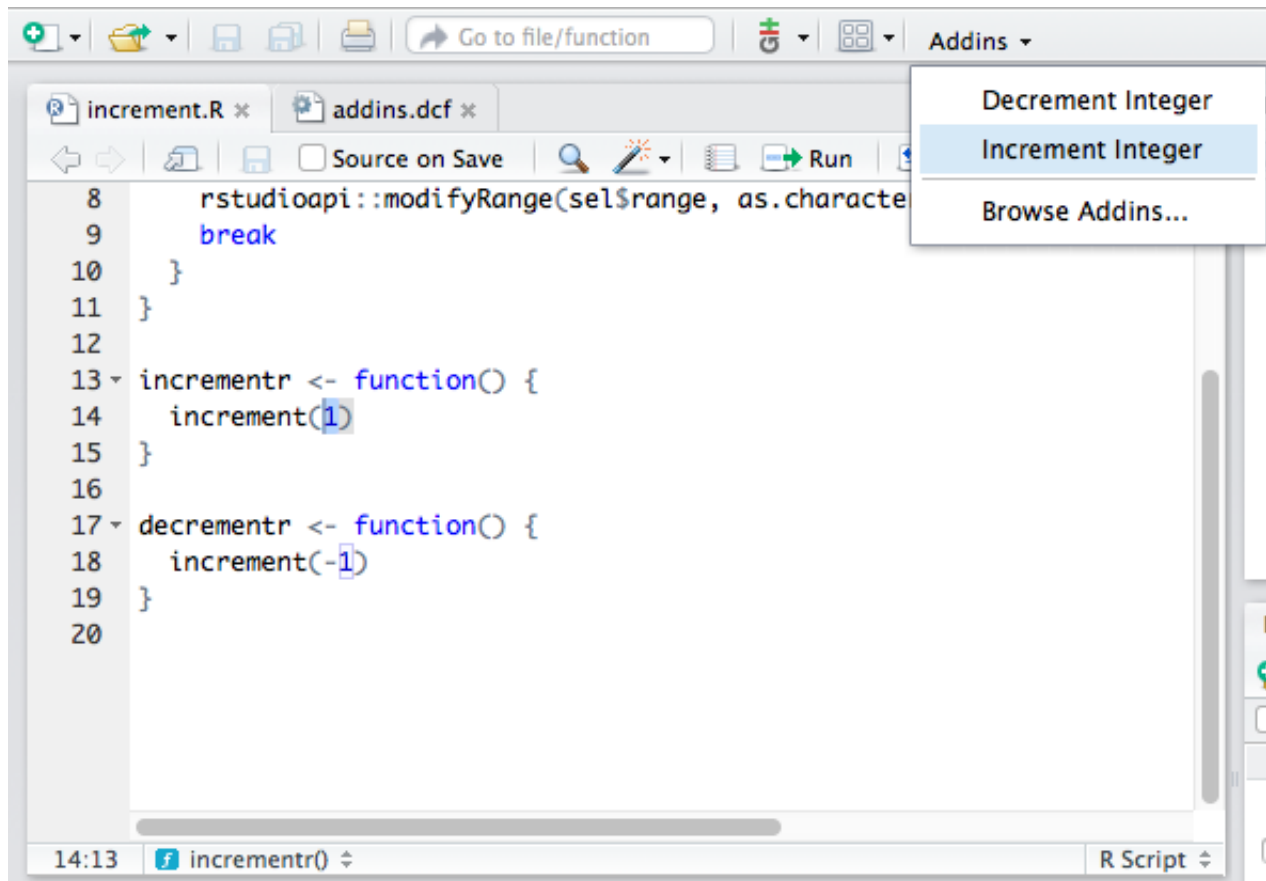
Binding: decremenr

Interactive: false

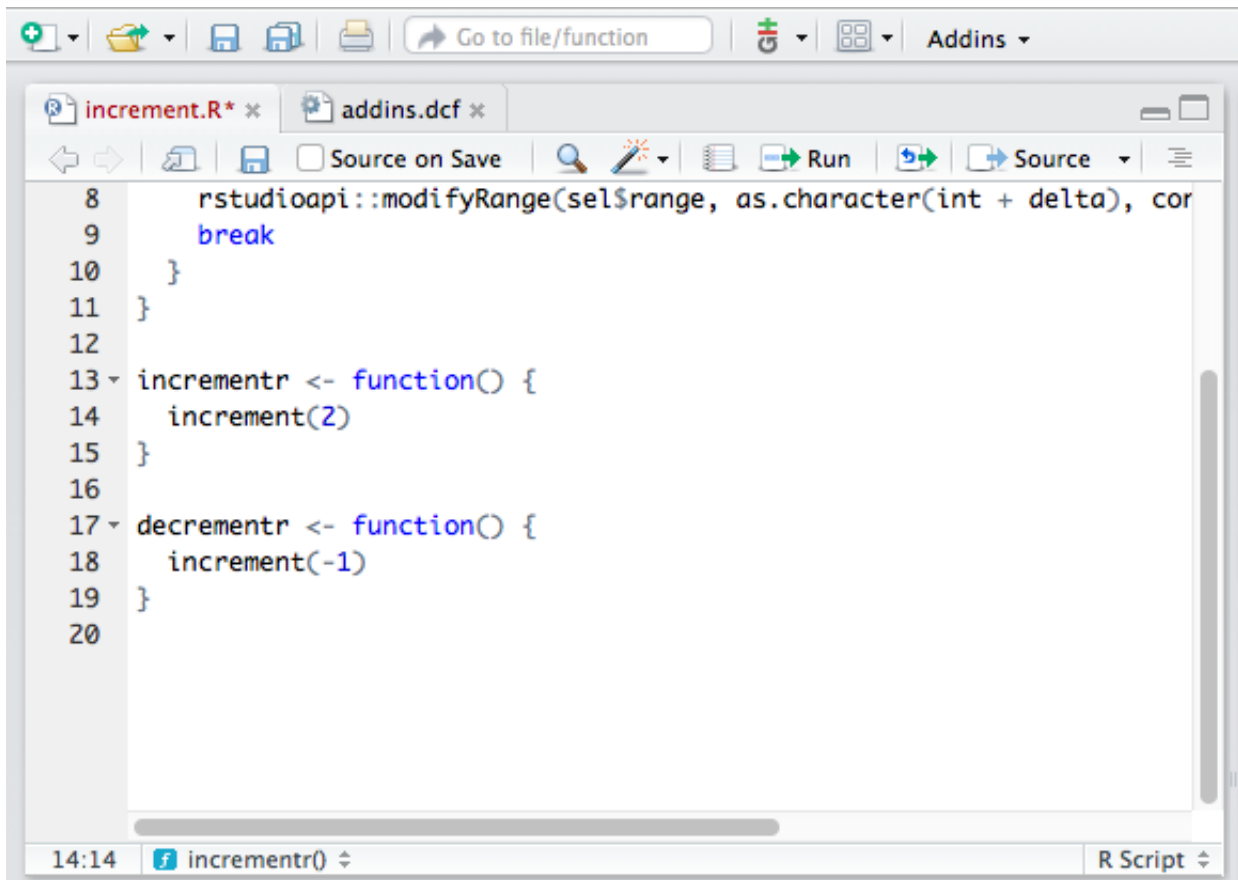
Build and install the package



Test the add-in



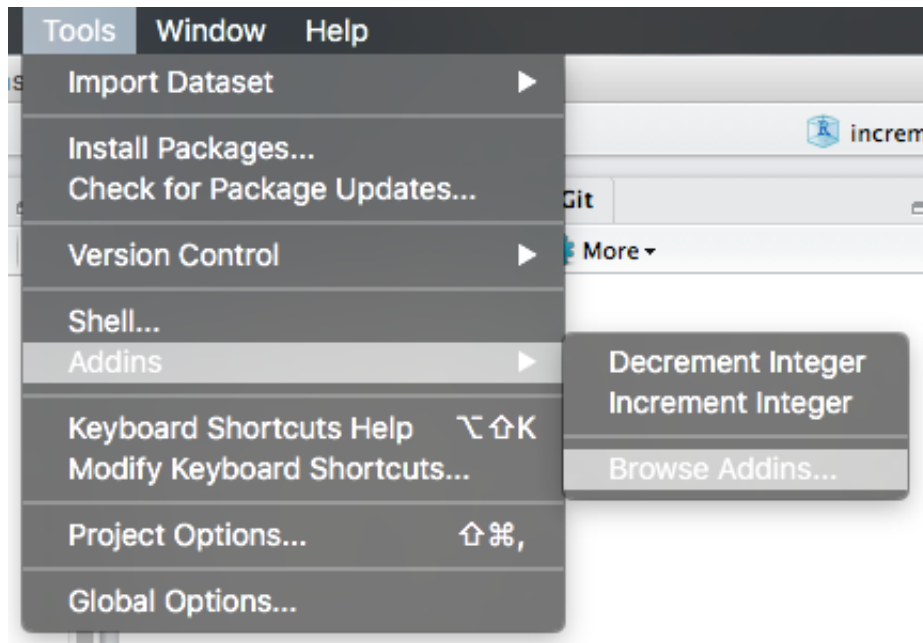
It works!



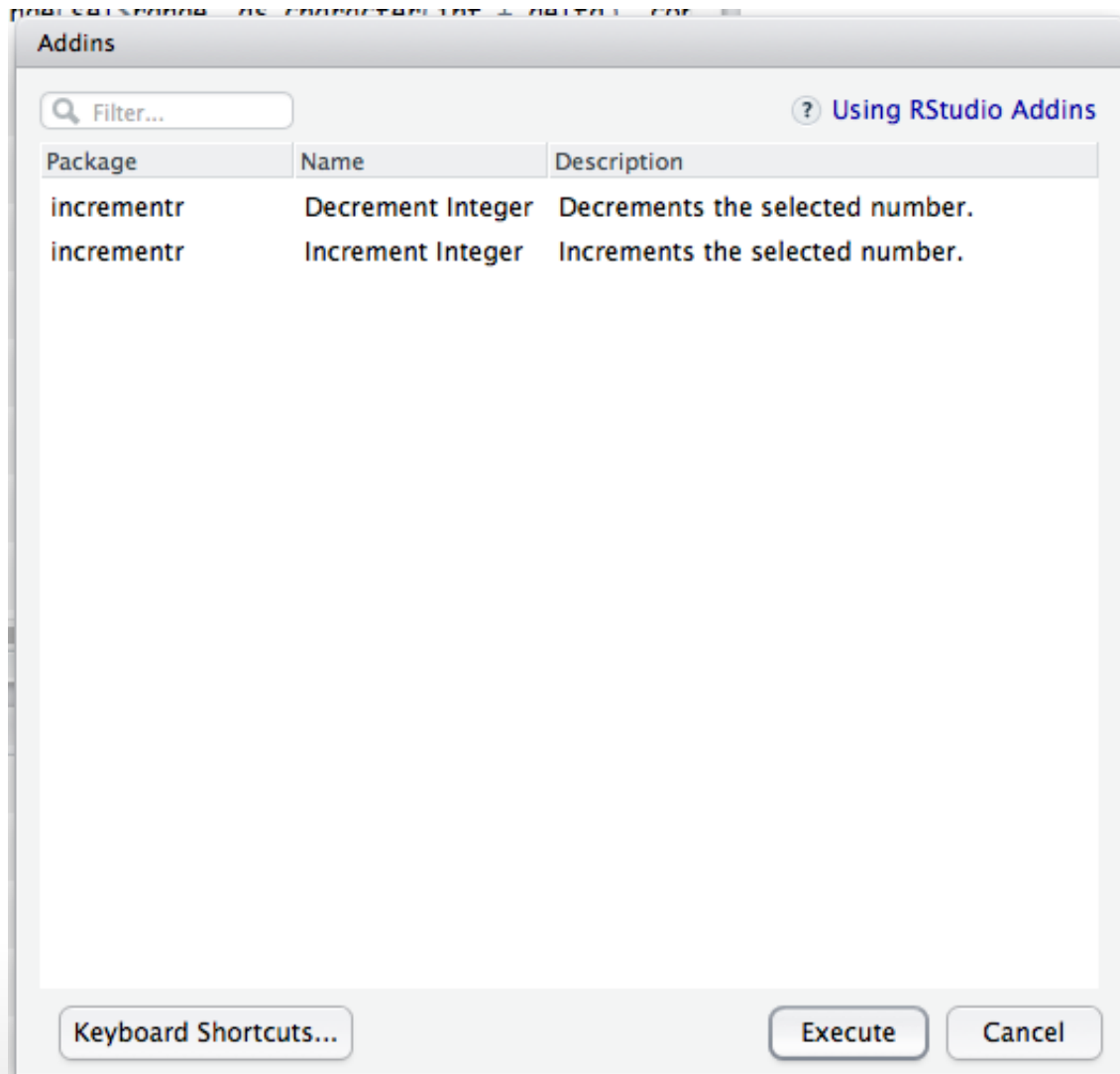
```
8     rstudioapi::modifyRange(sel$range, as.character(int + delta), cor
9     break
10  }
11  }
12
13  incrementr <- function() {
14    increment(2)
15  }
16
17  decremenr <- function() {
18    increment(-1)
19  }
20
```

The screenshot shows the RStudio interface. The top toolbar includes icons for file operations and a search bar. The editor window has two tabs: 'increment.R*' and 'addins.dcf'. The 'increment.R' tab is active, displaying the R code shown in the pre block. The status bar at the bottom indicates the current line is 14:14 and the function being edited is 'incrementr()'.

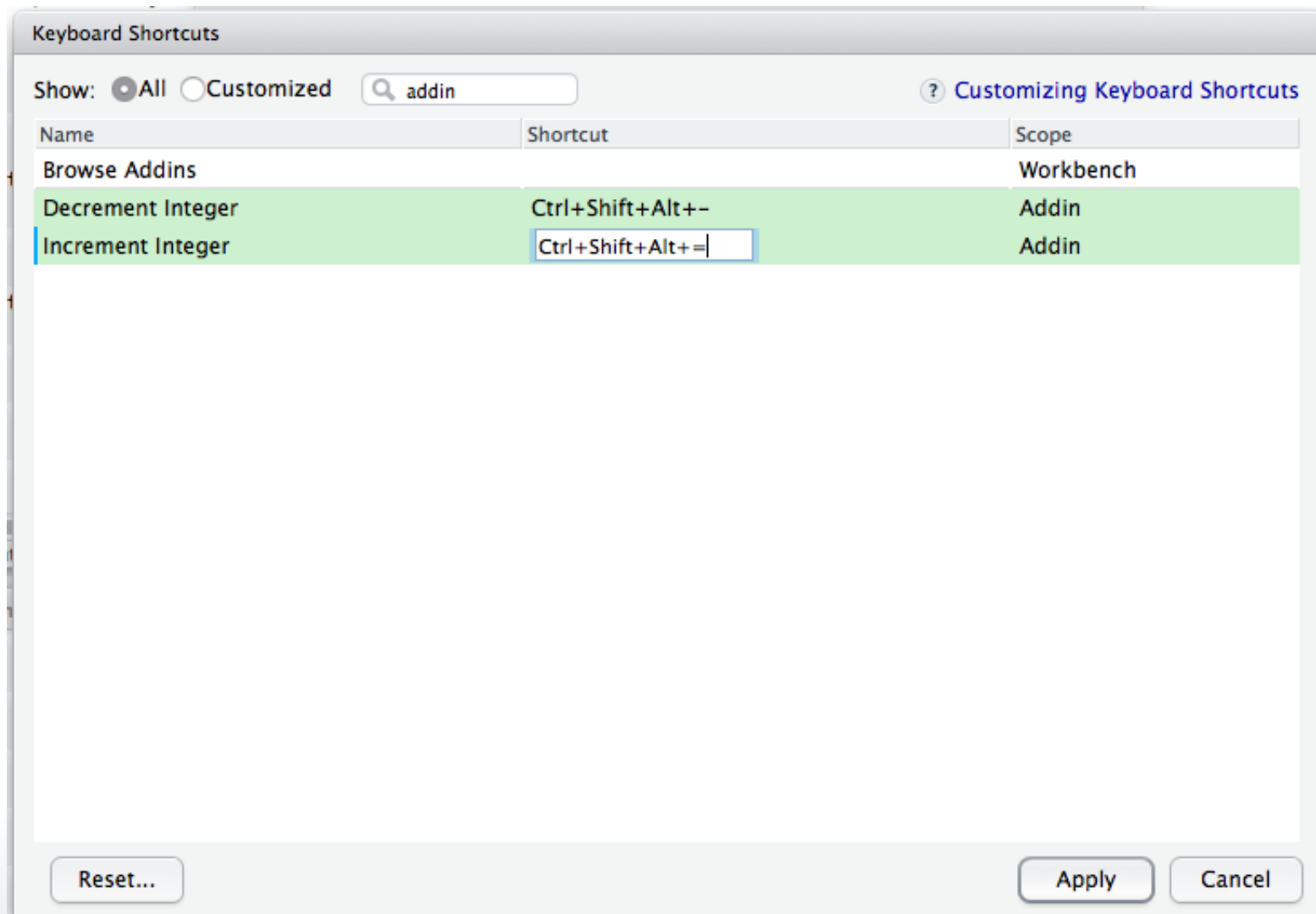
Binding a keyboard shortcut, step 1



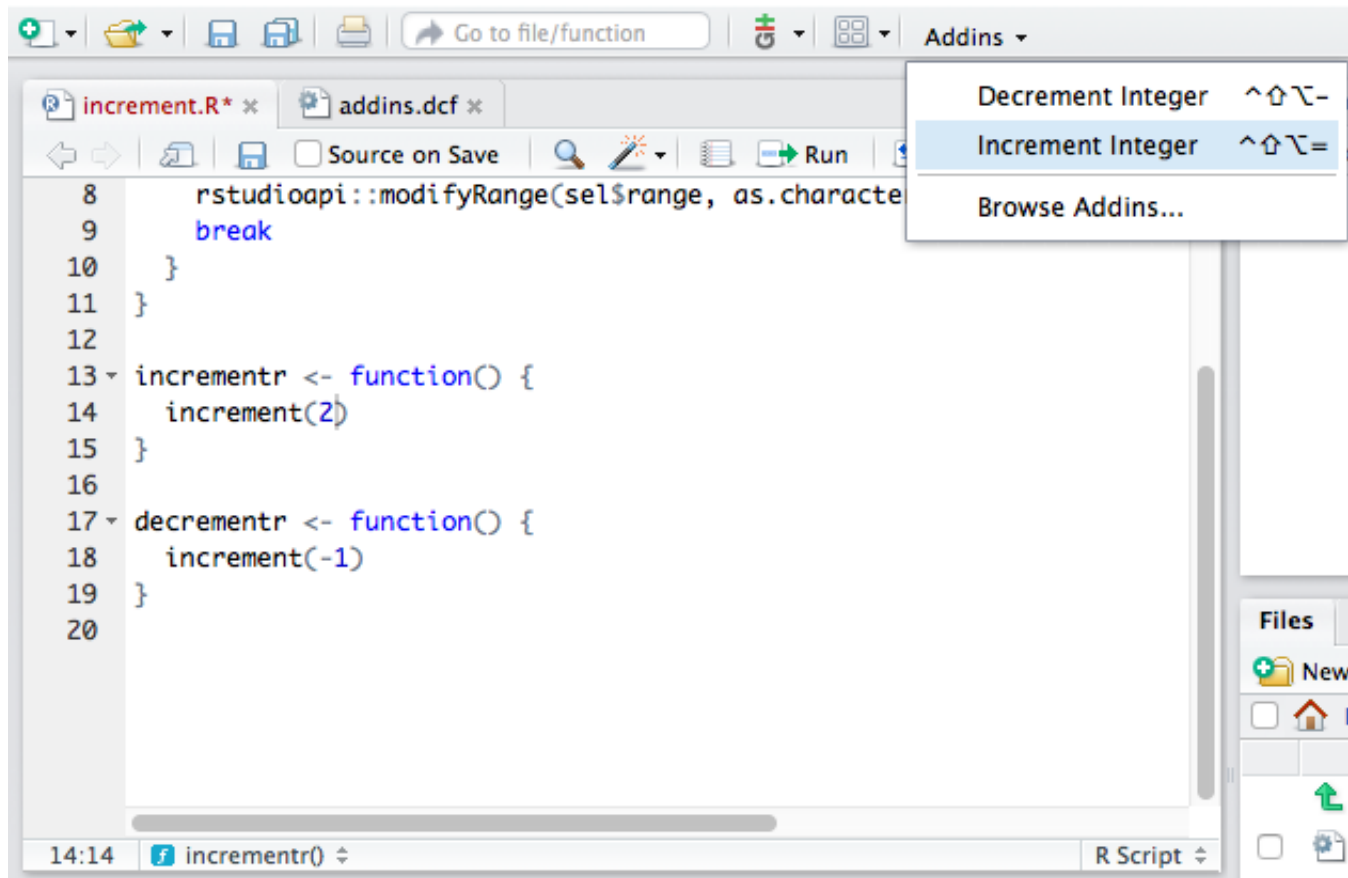
Binding a keyboard shortcut, step 2



Binding a keyboard shortcut, step 3



Add-in with keyboard shortcuts



Sharing an add-in

Share with existing mechanisms, or:

1. Post on GitHub
2. `devtools::install_github("you/yourpackage")`

Package library = add-in library

- Use site library to make add-ins available to all users on your team
- Install add-ins to your own library
- Use Packrat to create project-scoped add-ins

Interactive add-ins

Cancel

Subset a data.frame

Done

Data

mtcars

Subset Expression

Show 25 entries

Search:

mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear
21	6	160	110	3.9	2.62	16.46	0	1	4
21	6	160	110	3.9	2.875	17.02	0	1	4
22.8	4	108	93	3.85	2.32	18.61	1	1	4

Listening on http://127.0.0.1:4425

Compactly display the internal structure of an R object, a 31/34

Interactive add-ins

- More complex and powerful
- Use Shiny to build UI
- View in pane, dialog, or external window

Documentation

<https://rstudio.github.io/rstudioaddins/>

Q&A