

Using in VS Code

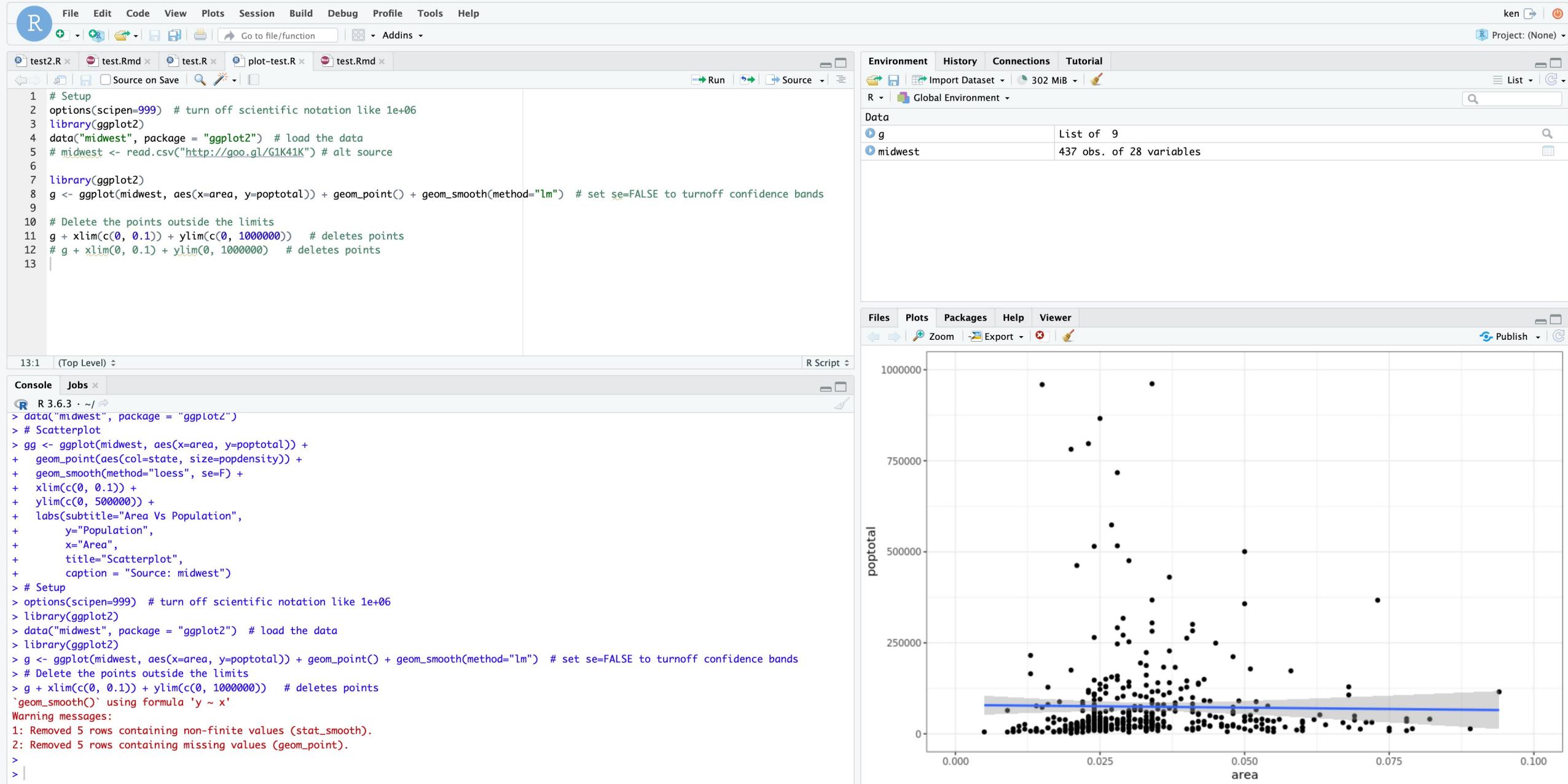
Kun Ren

renkun@outlook.com
github.com/renkun-ken

About me



- Hedge fund researcher
- Major collaborator of vscode-R and R language server
- Collaborator of data.table, lintr, etc.
- Author or contributor of numerous R packages on GitHub
- Microsoft MVP on Data Platform
- Author of *Learning R Programming*





RStudio Desktop/Server

Editor

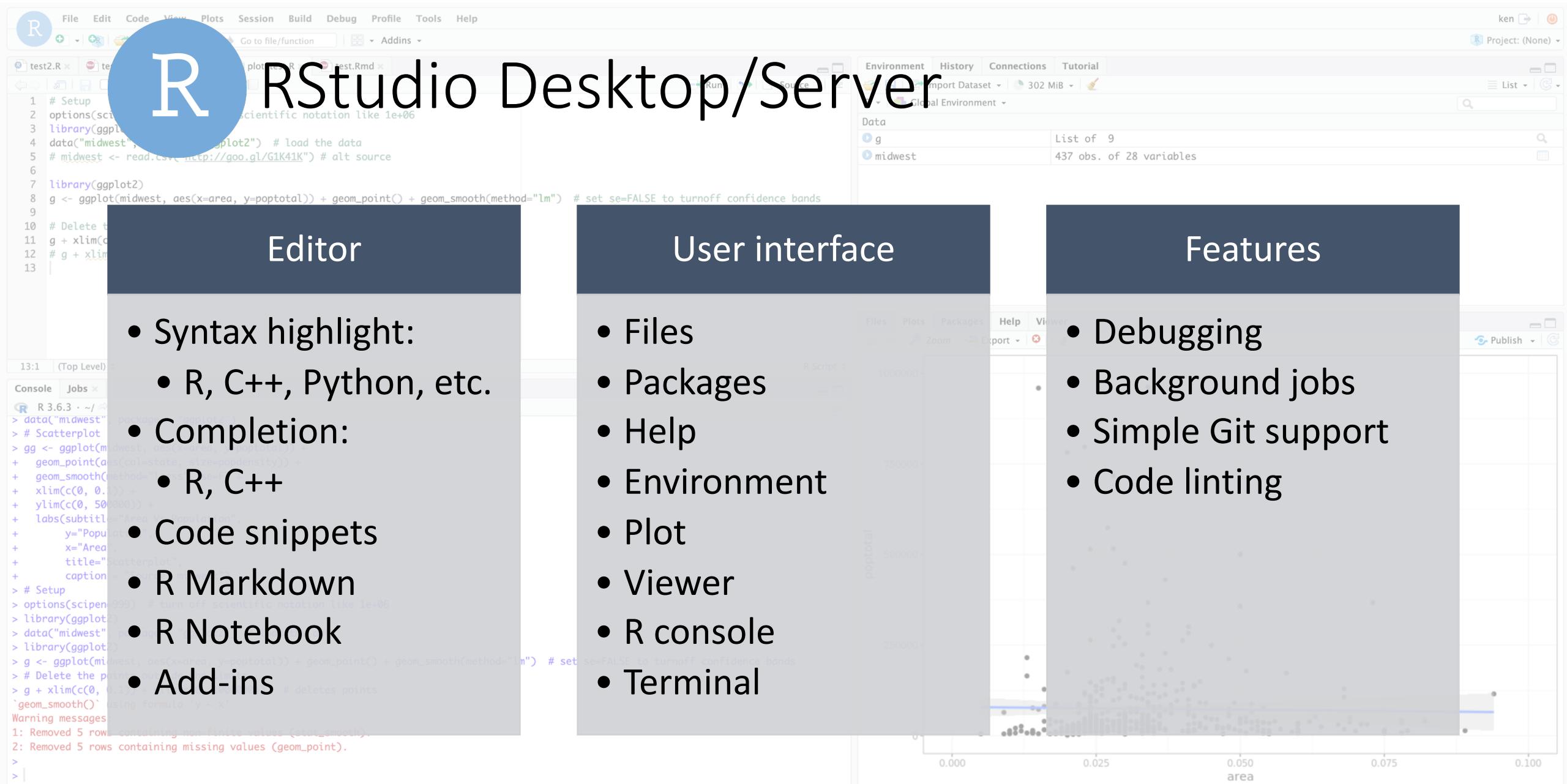
- Syntax highlight:
 - R, C++, Python, etc.
- Completion:
 - R, C++
- Code snippets
- R Markdown
- R Notebook
- Add-ins

User interface

- Files
- Packages
- Help
- Environment
- Plot
- Viewer
- R console
- Terminal

Features

- Debugging
- Background jobs
- Simple Git support
- Code linting



Great things about RStudio

- Just works out of the box
 - Intuitive all-in-one screen layout
 - Nice integration of packages: templates, add-ons, linting, etc.
- Tailored for the needs of a data scientist

```

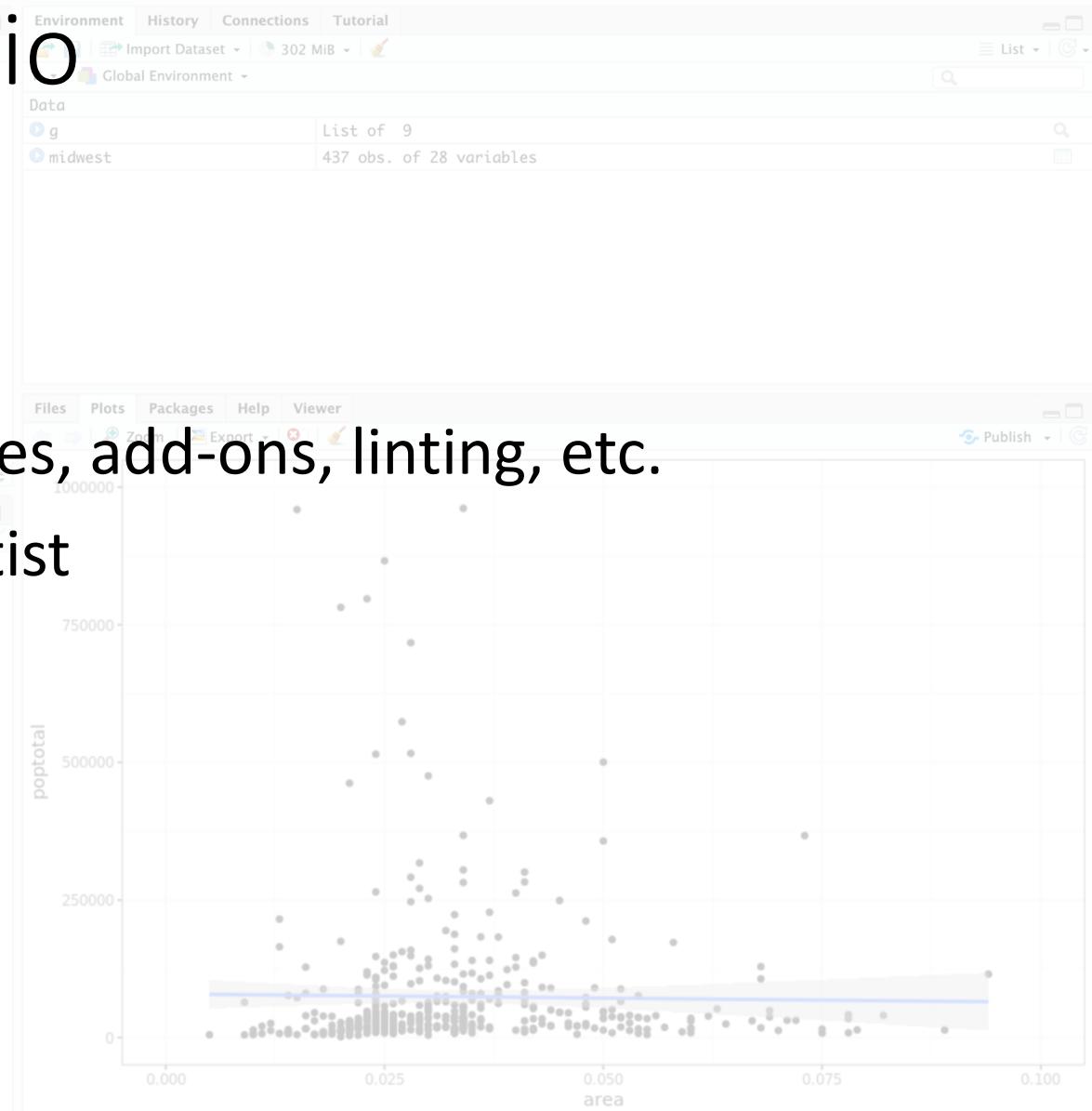
1: # Setup
2: options(scipen=999) # turn off scientific notation like 1e+06
3: library(ggplot2)
4: data("midwest", package = "ggplot2") # load the data
5: # midwest <- read.csv("http://goo.gl/G1K41K") # alt source
6:
7: library(ggplot2)
8: g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
9:
10: # Delete the points outside the limits
11: g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
12: # g + xlim(0, 0.1) + ylim(0, 1000000) # deletes points
13:

```

```

13:1 (Top Level) ◊
Console Jobs ×
R 3.6.3 · ~/Documents/RStudio
> data("midwest", package = "ggplot2")
> # Scatterplot
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(col=state, size=popdensity)) +
+   geom_smooth(method="loess", se=F) +
+   xlim(c(0, 0.1)) +
+   ylim(c(0, 500000)) +
+   labs(subtitle="Area Vs Population",
+        y="Population",
+        x="Area",
+        title="Scatterplot",
+        caption = "Source: midwest")
> # Setup
> options(scipen=999) # turn off scientific notation like 1e+06
> library(ggplot2)
> data("midwest", package = "ggplot2") # load the data
> library(ggplot2)
> g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
> # Delete the points outside the limits
> g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
`geom_smooth()` using formula 'y ~ x'
Warning messages:
1: Removed 5 rows containing non-finite values (stat_smooth).
2: Removed 5 rows containing missing values (geom_point).
>

```



But sometimes it is painful...

- Need to work with multiple interactive R sessions on one server
 - Across multiple projects
 - In the same project

The screenshot shows the RStudio interface with several windows open:

- Code Editor:** Shows a script named "test.R" with R code for data manipulation and plotting.
- Environment View:** Displays the global environment with objects "g" and "midwest".
- Plots View:** Shows a scatterplot of "poptotal" vs "area".
- Console View:** Displays the R command history, including the execution of the "test.R" script.

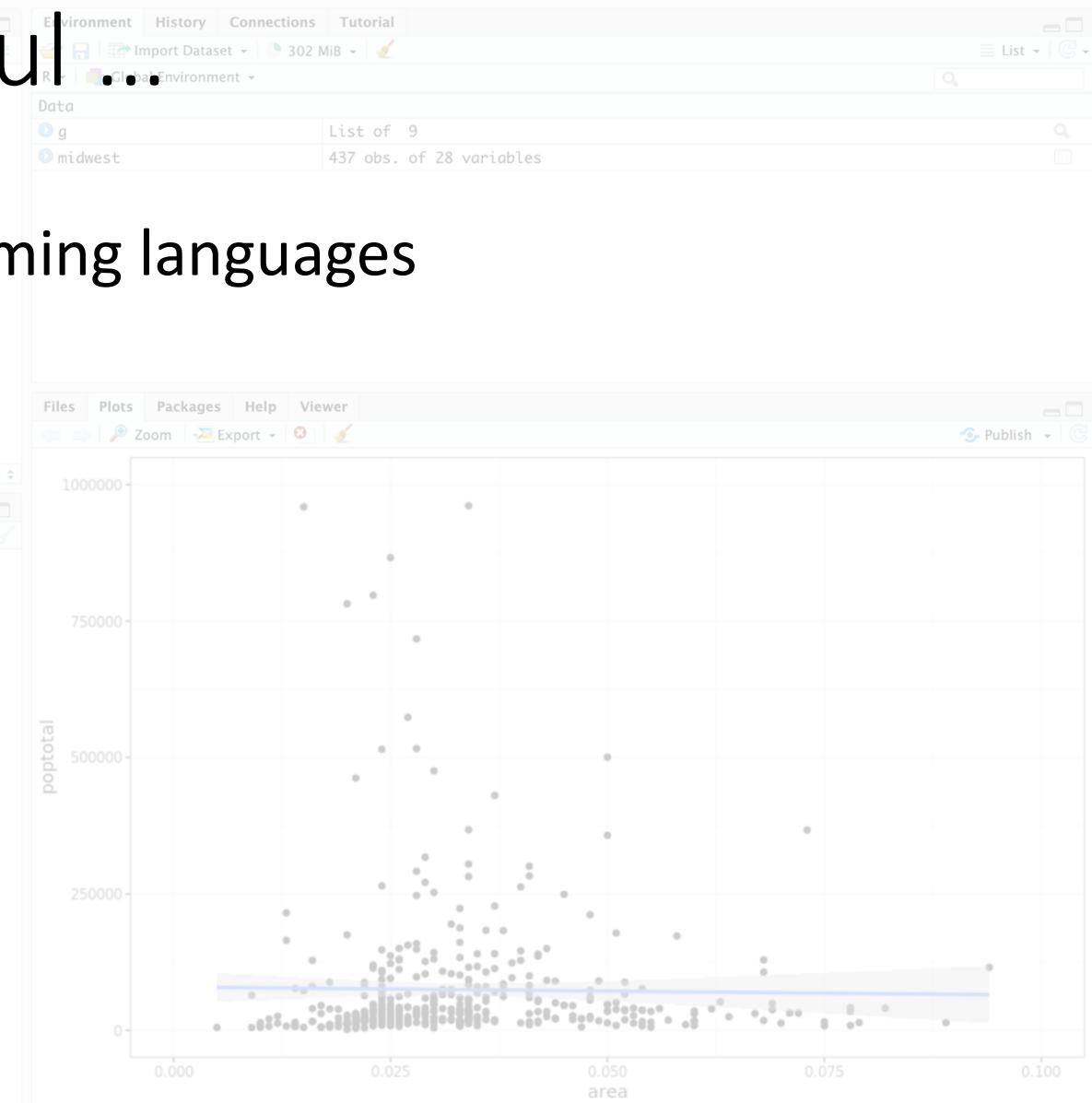
But sometimes it is painful...

- Need to work with multiple programming languages
 - C++ for better performance
 - R for easier programming and analysis
 - HTML/CSS/JavaScript for shiny apps

```

13:1 (Top Level) ◊
Console Jobs ×
R 3.6.3 · ~/Documents/R/ken/ken
> data("midwest", package = "ggplot2")
> # Scatterplot
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(col=state, size=popdensity)) +
+   geom_smooth(method="loess", se=F) +
+   xlim(c(0, 0.1)) +
+   ylim(c(0, 500000)) +
+   labs(subtitle="Area Vs Population",
+        y="Population",
+        x="Area",
+        title="Scatterplot",
+        caption = "Source: midwest")
> # Setup
> options(scipen=999) # turn off scientific notation like 1e+06
> library(ggplot2)
> data("midwest", package = "ggplot2") # load the data
> library(ggplot2)
> g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
> # Delete the points outside the limits
> g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
`geom_smooth()` using formula 'y ~ x'
Warning messages:
1: Removed 5 rows containing non-finite values (stat_smooth).
2: Removed 5 rows containing missing values (geom_point).
>

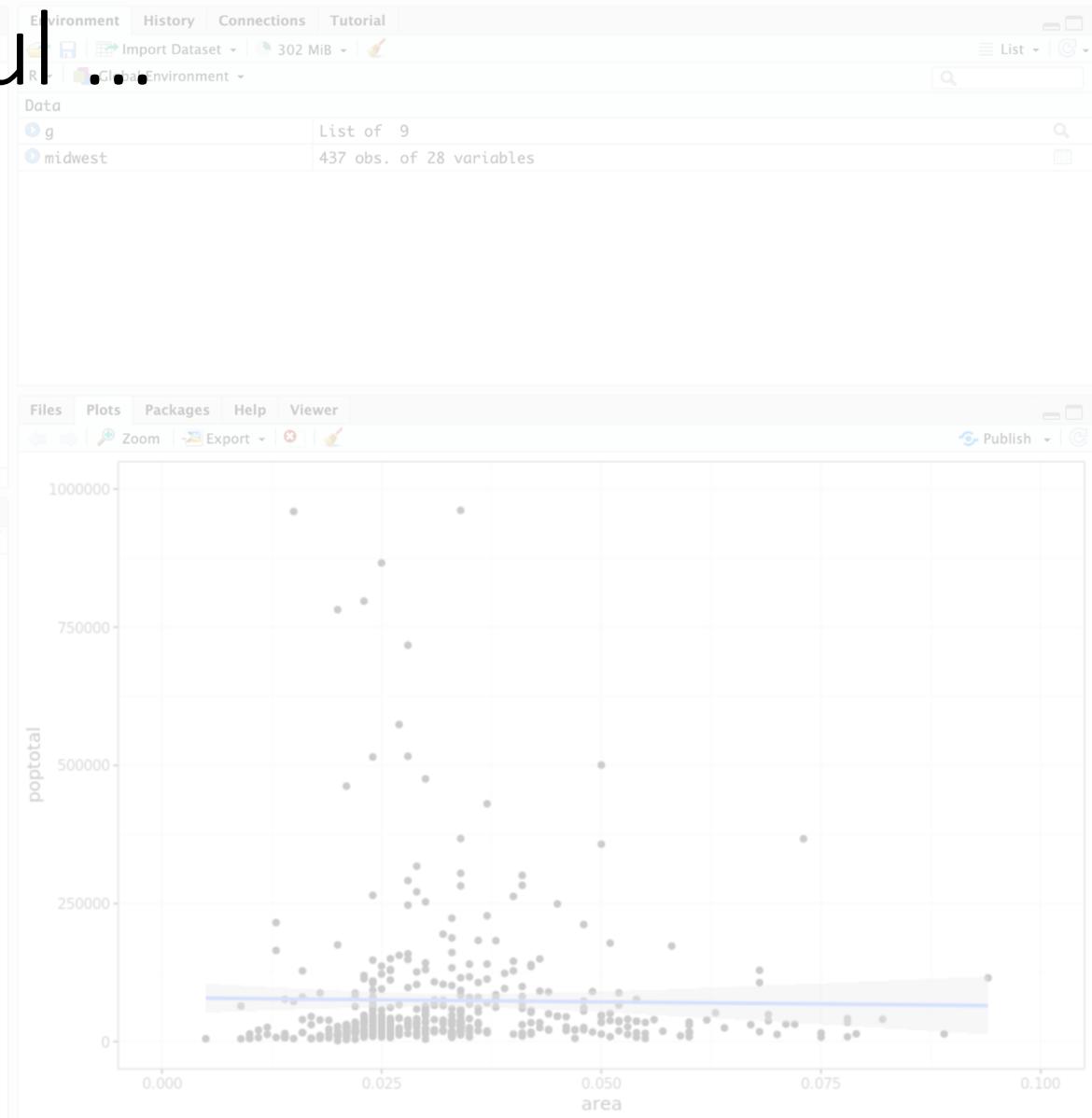
```



But sometimes it is painful...

- Need to work with multiple servers
 - Independent servers
 - HPC cluster

```
13:1 (Top Level) >
Console  Jobs <-
R 3.6.3 - ~/>
> data("midwest", package = "ggplot2")
> # Scatterplot
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(col=state, size=popdensity)) +
+   geom_smooth(method="loess", se=F) +
+   xlim(c(0, 0.1)) +
+   ylim(c(0, 500000)) +
+   labs(subtitle="Area Vs Population",
+        y="Population",
+        x="Area",
+        title="Scatterplot",
+        caption = "Source: midwest")
> # Setup
> options(scipen=999) # turn off scientific notation like 1e+06
> library(ggplot2)
> data("midwest", package = "ggplot2") # load the data
> library(ggplot2)
> g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
> # Delete the points outside the limits
> g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
`geom_smooth()` using formula 'y ~ x'
Warning messages:
1: Removed 5 rows containing non-finite values (stat_smooth).
2: Removed 5 rows containing missing values (geom_point).
>
```



R File Edit Code View Plots Session Build Debug Profile Tools Help

Project: (None)

But sometimes it is painful...

- Need to work with larger projects
 - Need more powerful engineering tools
 - Project-wide symbol search, rename, find references, ...

1 # Setup
 2 options(scipen=999) # turn off scientific notation like 1e+06
 3 library(ggplot2)
 4 data("midwest", package = "ggplot2") # load the data
 5 # midwest <- read.csv("http://goo.gl/GIK41K") # alt source
 6
 7 library(ggplot2)
 8 g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
 9
 10 # Delete the points outside the limits
 11 g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
 12 # g + xlim(0, 0.1) + ylim(0, 1000000) # deletes points
 13

13:1 (Top Level) :

Console Jobs

```
R 3.6.3 · ~/...> data("midwest", package = "ggplot2")
> # Scatterplot
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(col=state, size=popdensity)) +
+   geom_smooth(method="loess", se=F) +
+   xlim(c(0, 0.1)) +
+   ylim(c(0, 500000)) +
+   labs(subtitle="Area Vs Population",
+        y="Population",
+        x="Area",
+        title="Scatterplot",
+        caption = "Source: midwest")
> # Setup
> options(scipen=999) # turn off scientific notation like 1e+06
> library(ggplot2)
> data("midwest", package = "ggplot2") # load the data
> library(ggplot2)
> g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
> # Delete the points outside the limits
> g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
`geom_smooth()` using formula 'y ~ x'
Warning messages:
1: Removed 5 rows containing non-finite values (stat_smooth).
2: Removed 5 rows containing missing values (geom_point).
>
>
```

Environment History Connections Tutorial

Import Dataset 302 MB

Data

g List of 9
 midwest 437 obs. of 28 variables

Files Plots Packages Help Viewer

But sometimes it is painful...

```

1 # Setup
2 options(scipen=999) # turn off scientific notation like 1e+06
3 library(ggplot2)
4 data("midwest", package = "ggplot2") # load the data
5 # midwest <- read.csv("http://goo.gl/G1K41K") # alt source
6
7 library(ggplot2)
8 g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
9
10 # Delete the points outside the limits
11 g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
12 # g + xlim(0, 0.1) + ylim(0, 1000000) # deletes points
13

```

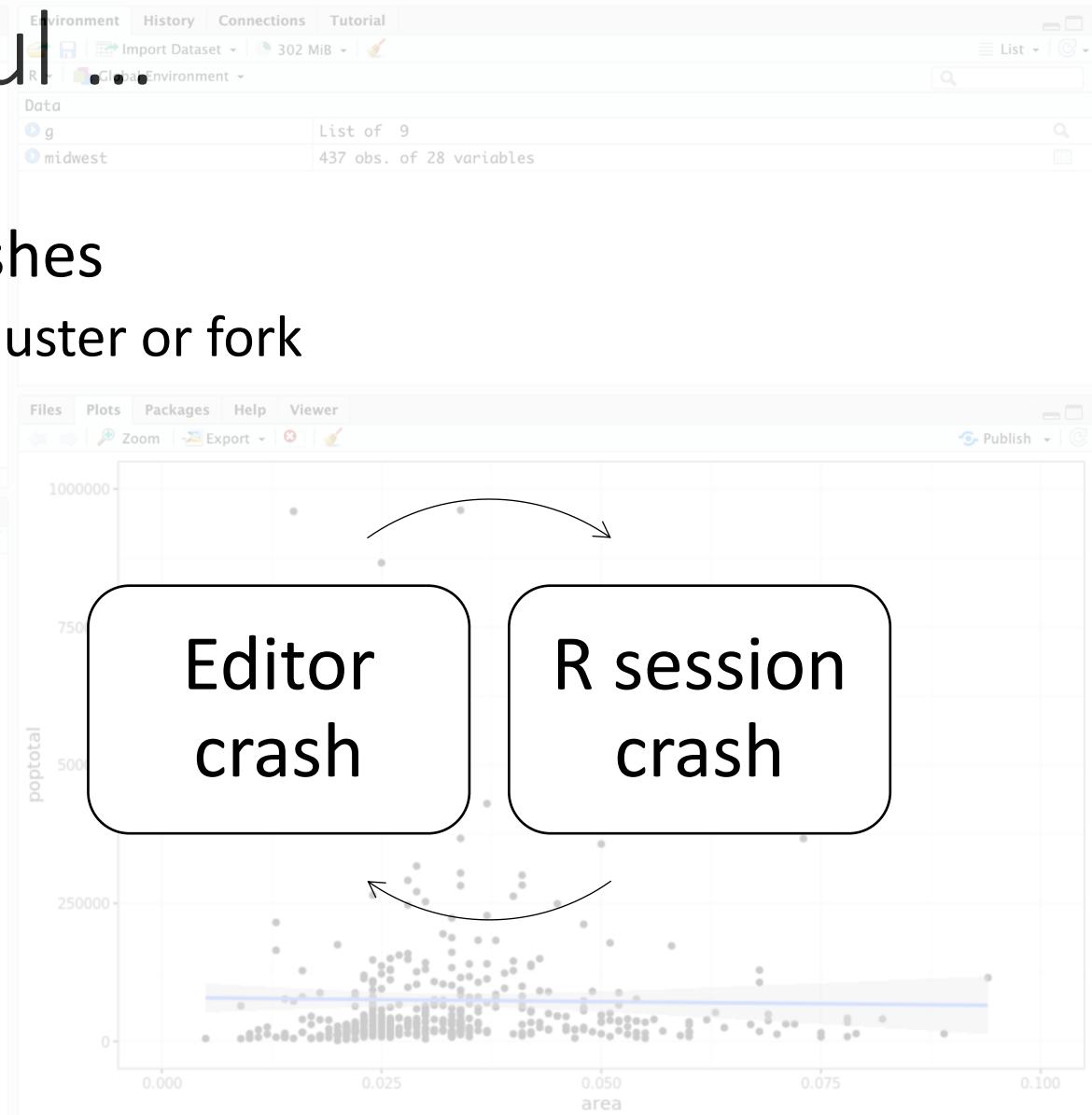
- Possible loss of code when editor crashes

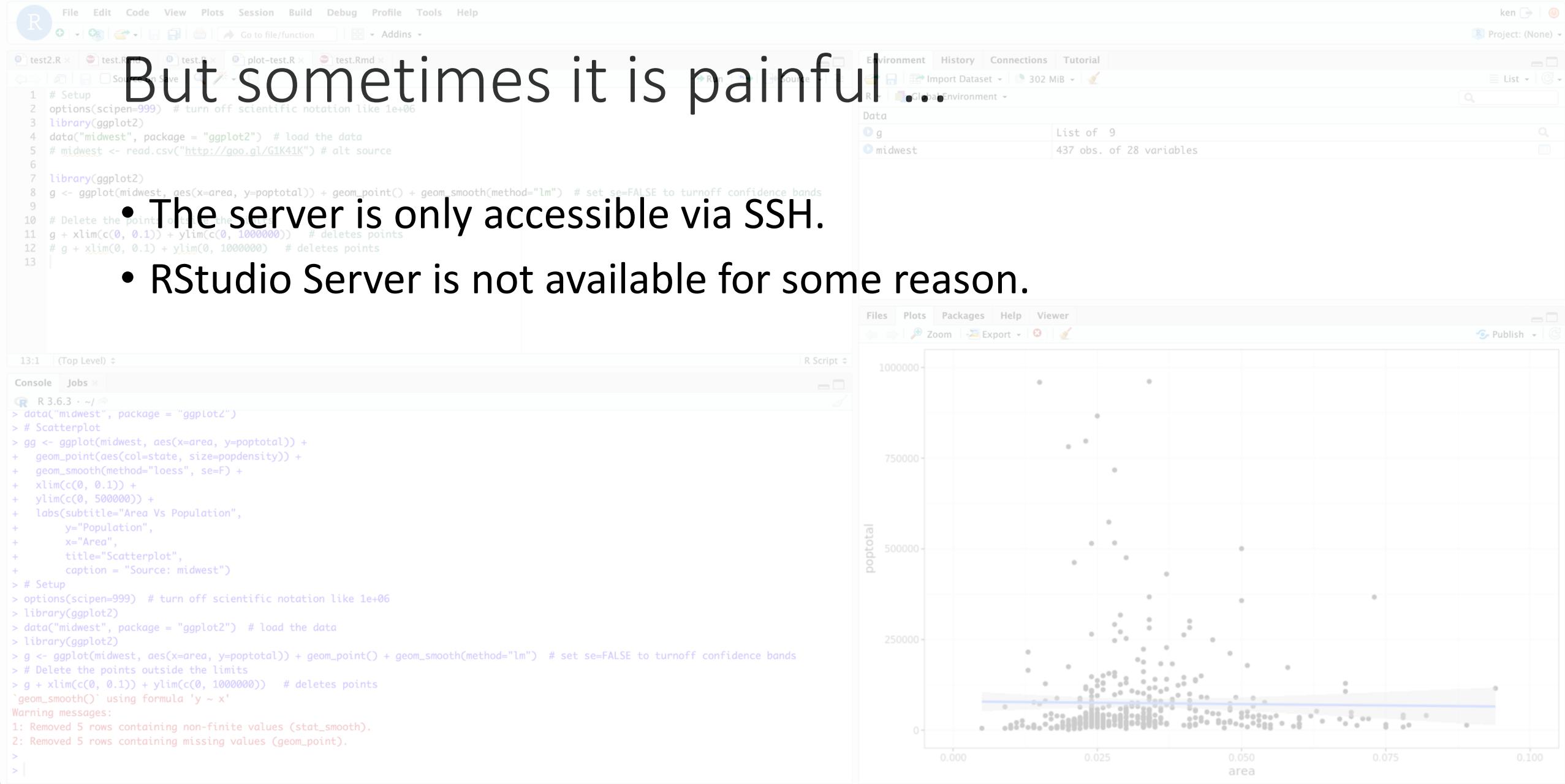
- Heavy parallel computing using socket cluster or fork
- C/C++ segfaults

```

13:1 (Top Level) ◊
Console Jobs ×
R 3.6.3 · ~/ ◊
> data("midwest", package = "ggplot2")
> # Scatterplot
> gg <- ggplot(midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(col=state, size=popdensity)) +
+   geom_smooth(method="loess", se=F) +
+   xlim(c(0, 0.1)) +
+   ylim(c(0, 500000)) +
+   labs(subtitle="Area Vs Population",
+        y="Population",
+        x="Area",
+        title="Scatterplot",
+        caption = "Source: midwest")
> # Setup
> options(scipen=999) # turn off scientific notation like 1e+06
> library(ggplot2)
> data("midwest", package = "ggplot2") # load the data
> library(ggplot2)
> g <- ggplot(midwest, aes(x=area, y=poptotal)) + geom_point() + geom_smooth(method="lm") # set se=FALSE to turnoff confidence bands
> # Delete the points outside the limits
> g + xlim(c(0, 0.1)) + ylim(c(0, 1000000)) # deletes points
`geom_smooth()` using formula 'y ~ x'
Warning messages:
1: Removed 5 rows containing non-finite values (stat_smooth).
2: Removed 5 rows containing missing values (geom_point).
>

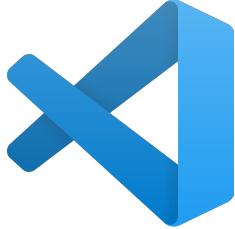
```







Visual Studio Code



Visual Studio Code

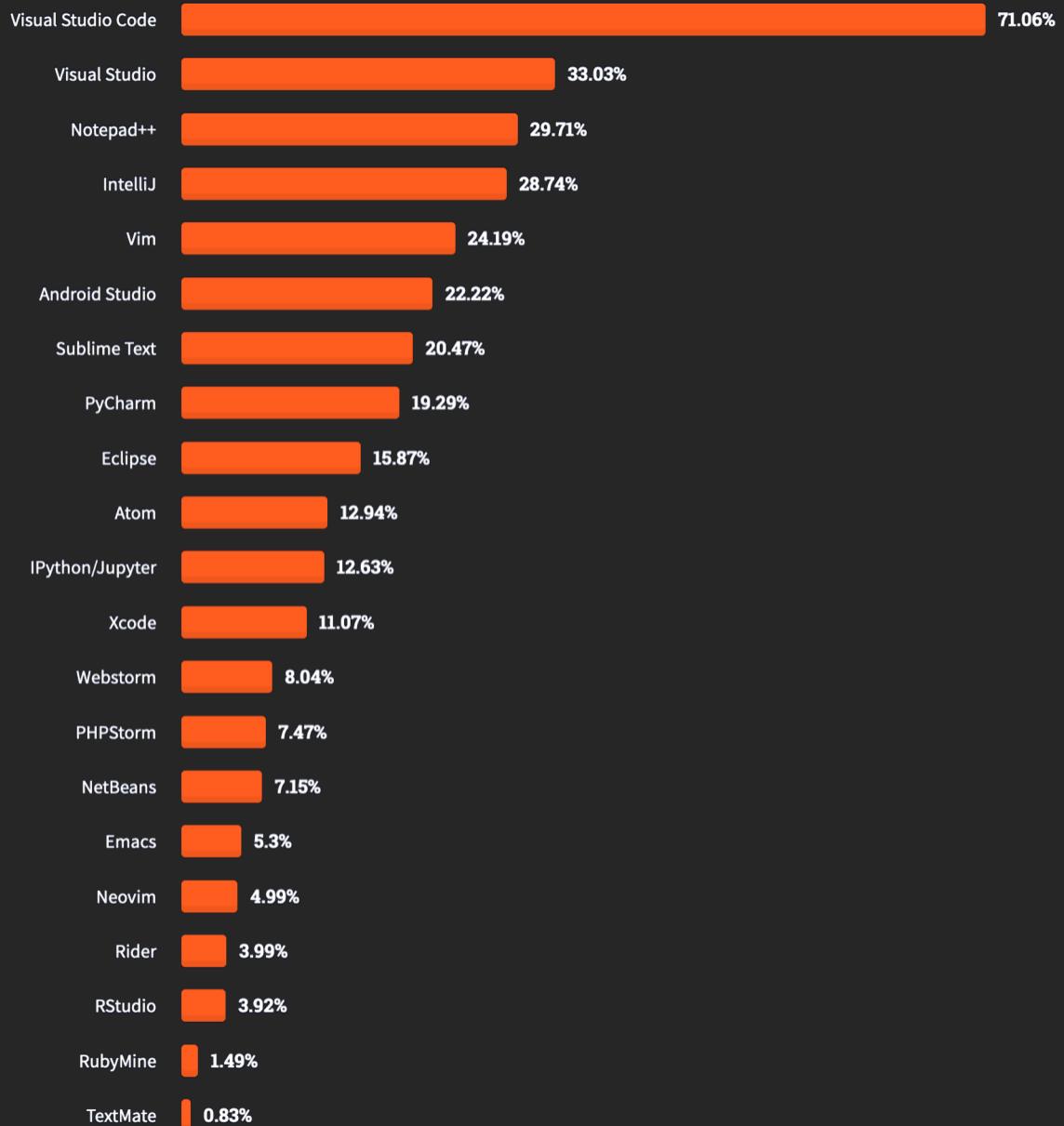
- Syntax highlight: many languages
- Language Server Protocol
- Debug Adapter Protocol
- Full-featured language supports: C/C++, C#, Java, Python, JavaScript, TypeScript, etc.
- Built-in Git support
- Tons of extensions
- Remote Development via SSH, WSL, and Docker
- Live collaboration

The screenshot shows the Visual Studio Code interface. On the left, the 'EXTENSIONS: MARKETPLACE' sidebar is open, displaying a list of installed extensions:

- Python 2019.6.24221 (by Microsoft) - Linting, Debugging (multi-threaded...)
- GitLens — Git supercharged 9.8.5 (by Eric Amodio) - Supercharge the Git capabilities built into VS Code
- C/C++ 0.24.0 (by Microsoft) - C/C++ IntelliSense, debugging, and build tools
- ESLint 1.9.0 (by Dirk Baeumer) - Integrates ESLint JavaScript into VS Code
- Debugger for Chrome 4.11.6 (by Microsoft) - Debug your JavaScript code in the browser
- Language Support for Java 0.47.0 (by Red Hat) - Java Linting, Intellisense, formattin...
- vscode-icons 8.8.0 (by VSCode Icons Team) - Icons for Visual Studio Code
- Vetur 0.21.1 (by Pine Wu) - Vue tooling for VS Code

The main code editor window shows a file named 'blog-post.js' from a 'gatsby-graphql-app' project. The code uses GraphQL queries and imports from 'gatsby-image'. The 'data' variable is highlighted with a tooltip, showing its type as 'Object'. The bottom status bar indicates the file is being compiled by 'Gatsby Develop'.

Stack Overflow Developer Survey 2021



A screenshot of a code editor interface showing an R script named `completion.R`. The editor has a dark theme with syntax highlighting for R code.

The code in the editor:

```
R > completion.R > scope_completion
329 scope_completion <- function(xdoc) {
330   if (is.null(xdoc)) {
331     return(list())
332   }
333
334   enclosing_scopes <- xdoc$xpath$find_all("//ancestor::*/@value")
335   point$row + 1, poi
336
337   You, 2 years ago · D
338   scope_symbol_nodes <- xml_find_all(enclosing_scopes, ".//value")
339   scope_symbol_names <- xml_text(scope_symbol_nodes)
340   scope_symbol_lines <- as.integer(xml_attr(scope_symbol_nodes, "line1"))
341   scope_symbol_selector <- match_with(scope_symbol_names, token)
342
343   scope_symbol_names <- rev(scope_symbol_names[scope_symbol_selector])
344   scope_symbol_lines <- rev(scope_symbol_lines[scope_symbol_selector])
345   scope_symbol_selector <- !duplicated(scope_symbol_names)
346
347   scope_symbol_names <- scope_symbol_names[scope_symbol_selector]
348   scope_symbol_lines <- scope_symbol_lines[scope_symbol_selector]
349
350   scope_symbol_completions <- .mapply(function(symbol, line) {
351     list(
352       label = symbol,
353       kind = CompletionItemKind$Field,
354       sortText = paste0(sort_prefixes$scope, symbol),
355       detail = "[scope]",
356       data = list(
357         type = "nonfunction",
358         uri = uri.
359     )
360   })
361
362   return(list(
363     completions = scope_symbol_completions,
364     line = line,
365     column = column
366   ))
367 }
```

The right panel displays detailed documentation for the `xml_text` function:

- xml_text {xml2}**: Extract or modify the text.
- Description**: `xml_text` returns a character vector, `xml_double` returns a numeric vector, `xml_integer` returns an integer vector.
- Usage**:
A snippet of R code demonstrating the usage of `xml_text`.

The bottom of the editor shows tabs for PROBLEMS, DEBUG CONSOLE, OUTPUT, and TERMINAL. The TERMINAL tab is active, showing the command `languageserver git:(master)`.

The left sidebar includes sections for EXPLORER, OPEN EDITORS, LANGUAGESERVER, OUTLINE, and TIMELINE. The LANGUAGESERVER section lists various completion item kinds and workspace symbols.

The bottom status bar shows the current file is `completion.R`, the status is `(not attached)`, and the date is `You, 2 years ago`.



completion.R (4724e42) ← completion.R (b2148e3)

Users > ken > Workspaces > github > languageserver > R > completion.R

```
308 imported_completions ← imported_object_compl
309 completions ← c(completions, imported_comple
310 }
311
312 completions
313 }
314
315 scope_completion_symbols_xpath ← paste(
316-  "[self::FUNCTION or self::OP-LAMBDA]/following-s
317  "forcond/SYMBOL",
318-  "*/LEFT_ASSIGN[not(following-sibling::expr/*[self
319-  "*/RIGHT_ASSIGN[not(preceding-sibling::expr/*[sel
320-  "*/EQ_ASSIGN[not(following-sibling::expr/*[self:
321  sep = "|")
322
323 scope_completion_funcs_xpath ← paste(
324-  "*/LEFT_ASSIGN[following-sibling::expr/*[self::FU
325-  "*/RIGHT_ASSIGN[preceding-sibling::expr/*[self::F
326-  "*/EQ_ASSIGN[following-sibling::expr/*[self::FUNC
327  sep = "|")
328
329 scope_completion ← function(uri, workspace, token, p
330  xdoc ← workspace$get_parse_data(uri)$xml_doc
331  if (is.null(xdoc)) {
332  |  return(list())
333  }
334
335 enclosing_scopes ← xdoc_find_enclosing_scopes(xd
336  point$row + 1, point$col + 1)
337
```

308 imported_completions ← imported_object_compl
309 completions ← c(completions, imported_comple
310 }
311
312 completions
313 }
314 Randy Lai, 4 years ago • new completion engine
315 scope_completion_symbols_xpath ← paste(
316+ "(*|descendant-or-self::exprlist/*)[self::FUNCTION
317+ "(*|descendant-or-self::exprlist/*)/LEFT_ASSIGN[n
318+ "(*|descendant-or-self::exprlist/*)/RIGHT_ASSIGN[
319+ "(*|descendant-or-self::exprlist/*)/EQ_ASSIGN[no
320 "forcond/SYMBOL",
321 sep = "|")
322
323 scope_completion_funcs_xpath ← paste(
324+ "(*|descendant-or-self::exprlist/*)/LEFT_ASSIGN[f
325+ "(*|descendant-or-self::exprlist/*)/RIGHT_ASSIGN[
326+ "(*|descendant-or-self::exprlist/*)/EQ_ASSIGN[fo
327 sep = "|")
328
329 scope_completion ← function(uri, workspace, token, p
330 xdoc ← workspace\$get_parse_data(uri)\$xml_doc
331 if (is.null(xdoc)) {
332 | return(list())
333 }
334
335 enclosing_scopes ← xdoc_find_enclosing_scopes(xd
336 point\$row + 1, point\$col + 1)
337



VS Code Marketplace

Extensions for the Visual Studio family of products

Search Visual Studio Code extensions 

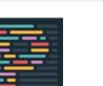
Featured

 GitHub Codespaces GitHub <small>635K</small>  FREE	 GitHub Classroom GitHub <small>10.1K</small>  FREE	 Bloop bloop <small>3.2K</small>  FREE	 Pester Tests Pester <small>1.6K</small>  FREE	 Node Dependencies Kasper Mikiewicz <small>1.3K</small>  FREE	 Type4Py Software Analytics <small>516</small>  FREE
--	--	---	--	--	---

Trending this week 

 FHIR Shorthand MITRE-Health <small>561</small>  FREE	 Terraform Betajob <small>127</small>  FREE	 Modern Fortran Giannis Nikiteas <small>226</small>  FREE	 PHP IntelliSense Damjan Cvetko <small>2K</small>  FREE	 .NET Watch Attach Trottero <small>229</small>  FREE	 Naruto Theme Bhargavi Chada <small>280</small>  FREE
--	--	--	---	---	--

Most Popular

 Python Microsoft <small>44.4M</small>  FREE	 C/C++ Microsoft <small>24.5M</small>  FREE	 Jupyter Microsoft <small>24.5M</small>  FREE	 Pylance Microsoft <small>17.2M</small>  FREE	 ESLint Dirk Baumer <small>17M</small>  FREE	 Prettier - Code formatter Prettier <small>16.1M</small>  FREE
---	--	--	---	---	---

<https://marketplace.visualstudio.com>

R

WORKSPACE

```
> g List of 9
> gg List of 9
> midwest tbl_df: 437 obs. of 28 variables
> midwest_select tbl_df: 6 obs. of 28 variables
> mpg tbl_df: 234 obs. of 11 variables
```

Scatterplot

```
gg <- ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col = state, size = popdensity)) +
  geom_smooth(method = "loess", se = FALSE) +
  xlim(c(0, 0.1)) +
  ylim(c(0, 500000)) +
  labs(subtitle = "Area Vs Population",
       y = "Population",
       x = "Area",
       title = "Scatterplot",
       caption = "Source: midwest")
```

plot(gg)

Need help getting started? Try the R Graphics Cookbook: <https://r-graphics.org>

geom_smooth() using formula 'y ~ x'

Warning messages:

- 1: Removed 15 rows containing non-finite values (stat_smooth).
- 2: Removed 15 rows containing missing values (geom_point).

```
> options(scipen = 999)
> library(ggplot2)
> library(ggalt)
midwest_select <- midwest[midwest$poptotal > 350000 &
  midwest$poptotal <= 500000 &
  midwest$area > 0.01 &
  midwest$area < 0.1, ]
```

Plot

```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col = state, size = popdensity)) + # draw points
  geom_smooth(method = "loess", se = F) +
  xlim(c(0, 0.1)) +
  ylim(c(0, 500000)) + # draw smoothing line
  geom_encircle(aes(x = area, y = poptotal),
    data = midwest_select,
    color = "red",
    size = 2,
    expand = 0.08) + # encircle
  labs(subtitle = "Area Vs Population",
       y = "Population",
       x = "Area",
       title = "Scatterplot + Encircle",
       caption = "Source: midwest")
```

Registered S3 methods overwritten by 'ggalt':

- method from grid.draw.absoluteGrob ggplot2
- gridHeight.absoluteGrob ggplot2
- gridWidth.absoluteGrob ggplot2
- grobx.absoluteGrob ggplot2
- groby.absoluteGrob ggplot2
- geom_smooth() using formula 'y ~ x'

Warning messages:

- 1: Removed 15 rows containing non-finite values (stat_smooth).
- 2: Removed 15 rows containing missing values (geom_point).

```
> # load package and data
library(ggplot2)

> data(mpg, package = "ggplot2") # alternate source: "http://goo.gl/uEeRGu"

> theme_set(theme_bw()) # pre-set the bw theme.
```

Scatterplot with overlapping points

```
> g <- ggplot(mpg, aes(cty, hwy))

> # Scatterplot
g + geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(subtitle = "mpg: city vs highway mileage",
       y = "hwy",
       x = "cty",
       title = "Scatterplot with overlapping points",
       caption = "Source: midwest")
```

geom_smooth() using formula 'y ~ x'

LIVE SHARE CONTROLS

0 △ 0 ① 22 Kun Live Share

R: 76532 Ln 8, Col 1 Spaces: 2 UTF-8 LF R



- + vscode-R
- + languageserver + httpgd
- + radian
- + R Debugger
- + some other vscode extensions:
 - + Utility extensions like Path Autocomplete
 - + C/C++ extension to support Rcpp development and debugging
 - + ...

radian

A 21st century R console

By Randy Lai

<https://github.com/randy3k/radian>

```
R version 3.4.1 (2017-06-30) -- Single Candle
Platform: x86_64-apple-darwin16.7.0

r$> x <- c(1.3, 3.6, 7.8)

r$> f <- function(x) {
  "multiline support"
}

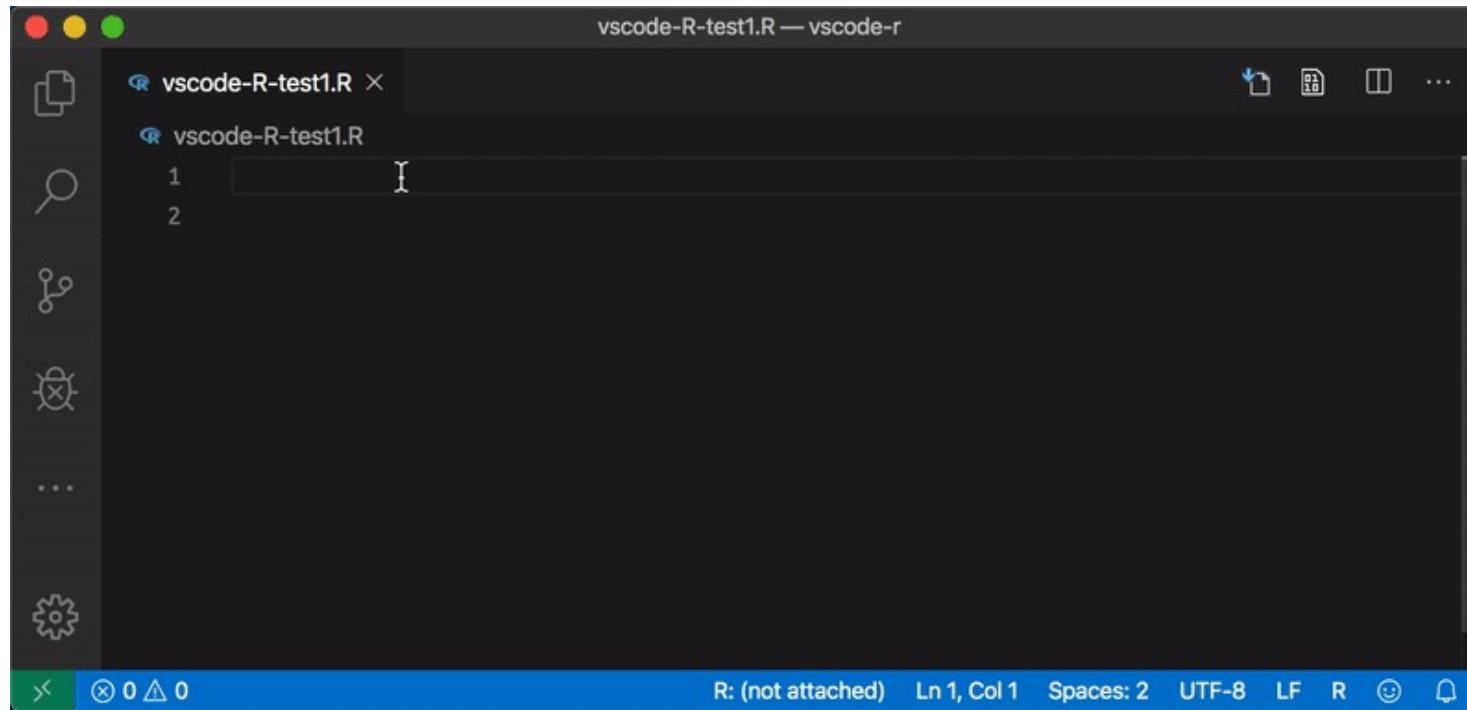
#!> echo "this is shell model"
this is shell model

r$> library(tidy)
      tidyverse
```

Code editing features

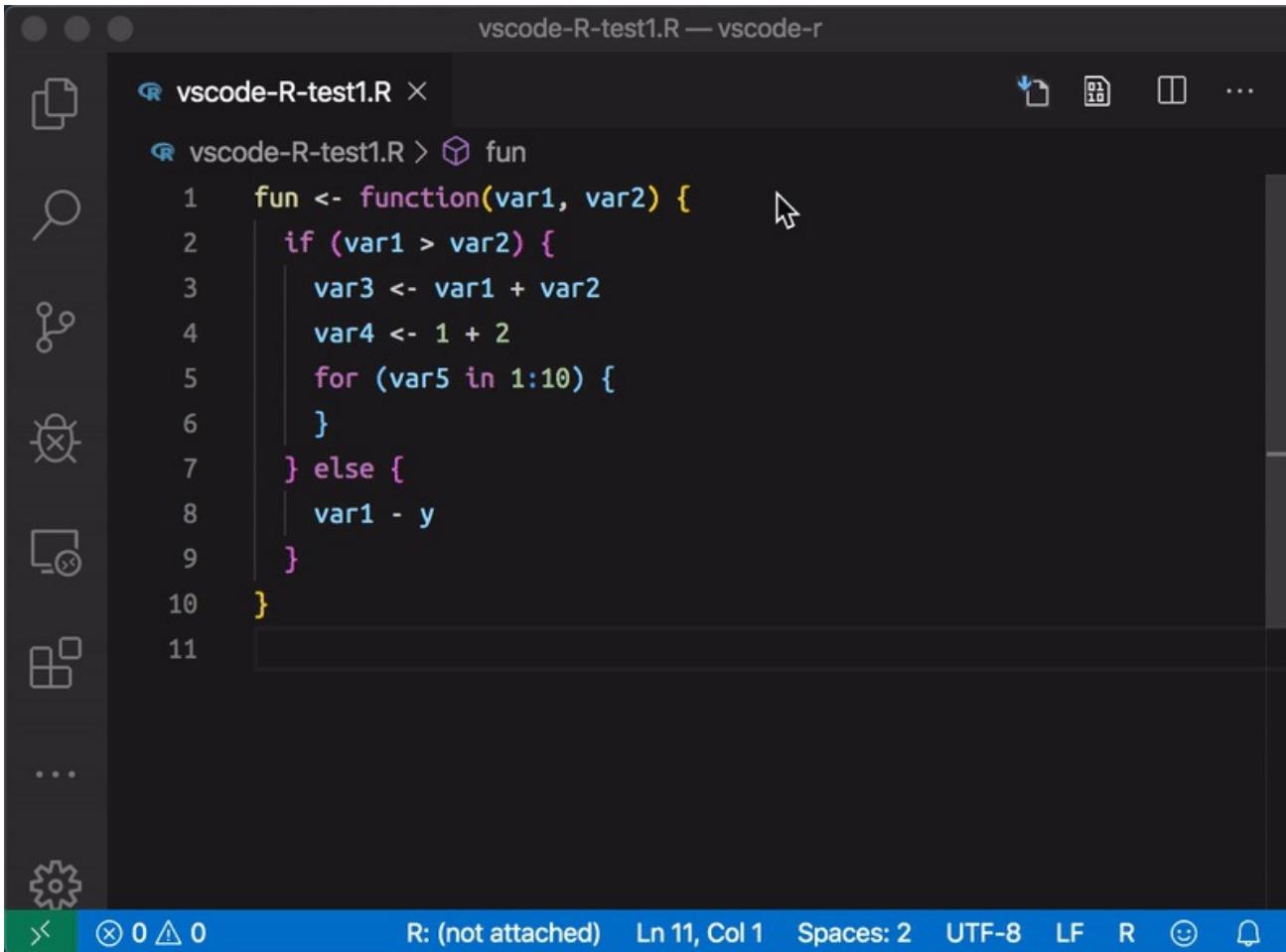
- Powered by R language server
- Based on static code analysis

Completion



- Langauge constants
- Function arguments
- Package names
- Exported objects and functions
- Non-exported objects
- Scope symbols

Scope completion



A screenshot of the Visual Studio Code (VS Code) interface, specifically the R extension. The title bar says "vscode-R-test1.R — vscode-r". The left sidebar has icons for file, search, and other extensions. The main editor area shows the following R code:

```
1 fun <- function(var1, var2) {      ↗
2   if (var1 > var2) {
3     var3 <- var1 + var2
4     var4 <- 1 + 2
5     for (var5 in 1:10) {
6       }
7     } else {
8       var1 - y
9     }
10 }
```

The cursor is at the end of the first closing brace on line 10. A vertical line of small dots highlights the scope of the variable "var1" from its declaration in line 2 to its final value assignment in line 9. The status bar at the bottom shows "R: (not attached)" and "Ln 11, Col 1".

- Function argument
- Local variables
- for loop symbol

Document highlight

- Highlight selected symbols of the same type in document
- Quickly locate all occurrences of a symbol or token.

The screenshot shows the Visual Studio Code interface with a dark theme. On the left, the Document Outline sidebar is visible, displaying icons for files, search, symbols, and other navigation options. The main area is the code editor, which contains the following R code:

```
1  obs <- 100
2  x <- rnorm(obs, mean = 1, sd = 1)
3  y <- 2 * x + rnorm(obs)
4  w <- runif(obs)
5  m <- lm(y ~ x, weights = w)
6  coef(m)
7
8  result <- local({
9    x <- rnorm(obs)
10   y <- rnorm(obs)
11   (x + y) * w
12 })
13
14 fun <- function(var1, var2) {
15   if (var1 > var2) {
16     var3 <- var1 + var2
17     var4 <- 1 + 2
18     for (var5 in 1:10) {
19       var1 + var2 + var3 + var5
20     }
21   } else {
22     var1 - y
23   }
24 }
25
26 fun(10, var2 = 20)
```

A cursor is positioned over the word "fun" in the 14th line of the code. The entire word "fun" is highlighted in yellow, demonstrating the "Document highlight" feature. The status bar at the bottom shows the file path "vscode-R-test1.R — vscode-r", line count "R: (not attached) Ln 13, Col 1", and other settings like "Spaces: 2", "UTF-8", "LF", and "R".

Diagnostics based on lintr

```
resu Put spaces around all infix operators. lintr
      x Peek Problem  No quick fixes available
      y <- rnorm(obs)
      (x + y) * w
    })
```

```
      (x + y) * w
    })
Commas should always have a space after. lintr
      Peek Problem  No quick fixes available
fun <- function(var1, var2) {
  if (var1 > var2) {
    var3 <- var1 + var2
    var4 <- 1 + 2
    for (var5 in 1:10) {
      var1 + var2 + var3 + var5
    }
  } else {
    var1 - y
  }
}

fun(10, var2 = 20)
```

Diagnostics based on lintr

```
338 | scope_symbol_nodes ← xml_find_all(enclosing_scopes, scope_completion_symbols_xpath)
339 | scope_symbol_names ← xml_text(scope_symbol_nodes)
340 | scope_symbol_lines ← as.integer(xml_attr(scope_symbol_nodes, "line1"))    Commas should always have a space after.
341 | scope_symbol_selector ← match_with(scope_symbol_names, token)
342 |
343 | scope_symbol_names ← rev(scope_symbol_names[scope_symbol_selector])
344 | scope_symbol_lines ← rev(scope_symbol_lines[scope_symbol_selector])
345 | scope_symbol_selector ← !duplicated(scope_symbol_names)    unexpected ')'
346 | You, a year ago • Show documentation for scope completion items
347 | scope_symbol_names ← scope_symbol_names[scope_symbol_selector]
348 | scope_symbol_lines ← scope_symbol_lines[scope_symbol_selector]
349 |
```

Code formatting

- Customizable formatting by styler
- Document formatting

The screenshot shows the R-test1.R file in the VS Code editor. The code uses syntax highlighting to distinguish between different R language elements:

- Variables: `obs`, `x`, `y`, `w`, `m`, `result`, `fun`, `var1`, `var2`, `var3`, `var4`, `var5`.
- Functions: `rnorm`, `runif`, `lm`, `coef`, `local`, `function`.
- Control structures: `if`, `for`.
- Comments: `#`.

The code itself is as follows:

```
1 obs <- 100
2 x <- rnorm(obs, mean=1, sd=1)
3 y <- 2*x+rnorm(obs)
4 w <- runif(obs)
5 m <- lm(y~x, weights = w)
6 coef(m)
7
8 result <- local({
9   x <-rnorm(obs)
10  y <-rnorm(obs)
11  (x+y)*w
12 })
13
14 fun <- function(var1,var2) {
15   if (var1>var2) {
16     var3 <- var1+var2
17     var4<-1+2
18     for (var5 in 1:10) {
19       var1+var2+var3+var5
20     }
21   } else {
22     var1-y
23   }
24 }
25
26 fun(10, var2 = 20)
27
```

Code formatting

- Range formatting
- On-type formatting

A screenshot of the Visual Studio Code (VS Code) interface, specifically showing an R script file named "on-type-formatting.R". The file contains the following code:

```
1 if (x>1) {  
2   if (y>1){  
3     x+y  
4   }  
5 }  
6
```

The word "if" is highlighted in pink, and the condition "x>1" is underlined in blue, indicating it is being edited. The cursor is positioned at the end of the second line. The status bar at the bottom shows the file is not attached, has 78 lines, and the current position is Line 6, Column 1. The status bar also indicates the encoding is UTF-8, and there are options for LF, R, and a gear icon.

Function Signature

```
1  obs <- 100
2  x <- rnorm(obs, mean = 1, sd = 2)
3  y <- 2 * x + rn
4  w <- runif(obs)
5  m <- lm(y ~ x,
6  coef(m) Density, distribution function, quantile function and random
7
8  result <- local({
9    x <- rnorm(obs)
10   y <- rnorm(obs)
11   (x + y) * w
12 })
```

generation for the normal distribution with mean equal to `mean` and standard deviation equal to `sd` .

Hover

- Show documentation for function on hover
- Show argument description for function argument on hover
- Show definition of symbols on hover
- Work with both package symbols and document symbols

```
vscode-R-test1.R — vscode-r

vscode-R-test1.R > ...
1  obs <- 100
2  x <- rnorm(obs, mean = 1, sd = 1)
3  y <- 2 * x + rnorm(obs)
4  w <- runif(obs)
5  m <- lm(y ~ x, weights = w)
6  coef(m)
7
8  result <- local({
9    x <- rnorm(obs)
10   y <- rnorm(obs)
11   (x + y) * w
12 })
13
14 fun <- function(var1, var2) {
15   if (var1 > var2) {
16     var3 <- var1 + var2
17     var4 <- 1 + 2
18     for (var5 in 1:10) {
19       var1 + var2 + var3 + var5
20     }
21   } else {
22     var1 - y
23   }
24 }
25
26 fun(10, var2 = 20)
27
```

R: (not attached) Ln 7, Col 1 Spaces: 2 UTF-8 LF R

Definition

- Function definition
- Symbol definition
- Quick preview
- Go to definition

The screenshot shows the VS Code interface with a dark theme. On the left is a sidebar with various icons: file, search, folder, refresh, document, and settings. The main area is a code editor titled "vscode-R-test1.R". The code is as follows:

```
1  obs <- 100
2  x <- rnorm(obs, mean = 1, sd = 1)
3  y <- 2 * x + rnorm(obs)
4  w <- runif(obs)
5  m <- lm(y ~ x, weights = w)
6  coef(m)
7
8  result <- local({
9    x <- rnorm(obs)
10   y <- rnorm(obs)
11   (x + y) * w
12 })
13
14 fun <- function(var1, var2) {
15   if (var1 > var2) {
16     var3 <- var1 + var2
17     var4 <- 1 + 2
18     for (var5 in 1:10) {
19       var1 + var2 + var3 + var5
20     }
21   } else {
22     var1 - y
23   }
24 }
25
26 fun(10, var2 = 20)
27
```

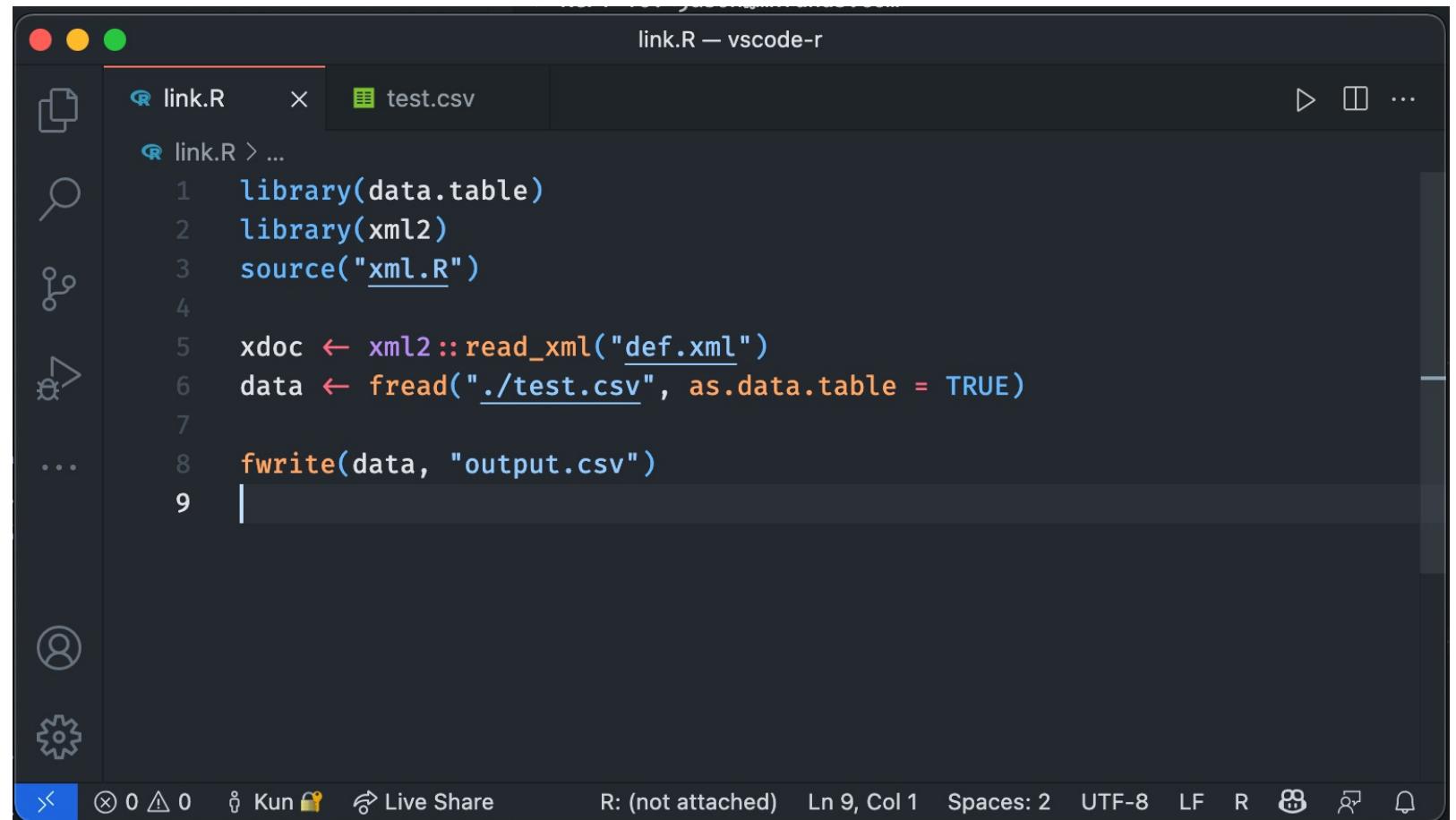
A cursor is positioned over the closing brace of the local assignment at line 12. A tooltip or preview window is visible above the cursor, showing the expanded code for the local assignment:

```
x <- rnorm(obs)
y <- rnorm(obs)
(x + y) * w
```

The status bar at the bottom indicates the file is not attached, has 7 lines and 1 column, uses spaces for indentation, and is in UTF-8 encoding.

Link

- Underlined path
recognized as
document links



The screenshot shows the VS Code interface with a dark theme. A sidebar on the left contains icons for file operations, search, and other settings. The main editor area has tabs for "link.R" and "test.csv". The "link.R" tab is active and displays the following R code:

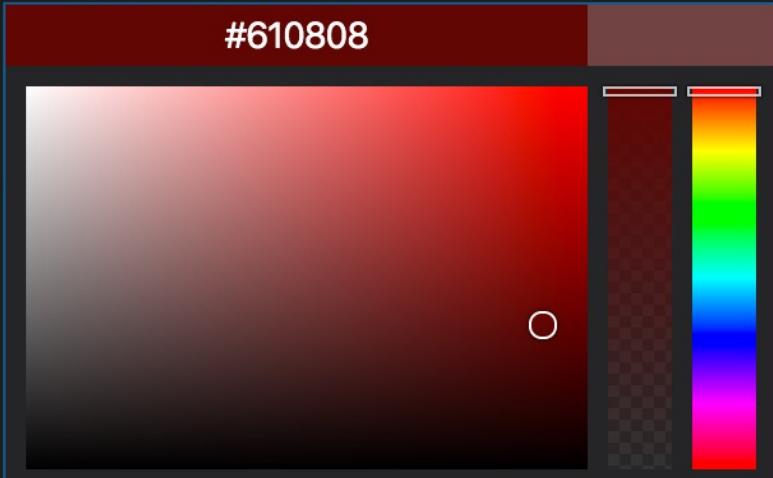
```
link.R > ...
1 library(data.table)
2 library(xml2)
3 source("xml.R")
4
5 xdoc ← xml2::read_xml("def.xml")
6 data ← fread("./test.csv", as.data.table = TRUE)
7
8 fwrite(data, "output.csv")
9 |
```

The code uses the `xml2` and `fread` functions from the `data.table` package. The paths `"def.xml"` and `"/test.csv"` are underlined, indicating they are recognized as document links.

Color

- Inline color preview
- Color picker

```
1  plot(rnorm(100), col = "#blue")
2  abline(h = 0, col = rgb(0.1, 0.5, 0.2))
3  grDevices::col2rgb(colors())
4
5  rgb(c(0.1, 0.2), c(0.1, 0.2), c(0.2, 0.1))
6
7  "#blue"
8
9  grDevices::col2rgb("#grey50")
10
11 "#blue"
12
13 col2rgb(c(
14
15 "#red"
16 "#tan1"
17 "#grey80"
18
19 col2rgb("##610808", alpha = TRUE)
```



Symbol navigation

- Document symbols
- Workspace symbols

The screenshot shows the RStudio IDE interface with the following details:

- EXPLORER View:** Shows a tree structure of open editors and language server files. The "interfaces.R" file is selected in the "OPEN EDITORS" section. In the "LANGUAGE SERVER" section, "interfaces.R" is also listed under "languagebase.R".
- Editors:** The main editor window displays the "interfaces.R" file content. A cursor is visible on the word "position" at line 11.
- Outline View:** Below the editor, the "OUTLINE" view is expanded, showing symbols related to "base interfaces". These include "document_uri", "location", "position", "print.document_uri", "print.location", "print.position", "print.range", "range", "symbol_information", and "text_document_position_params".
- Bottom Status Bar:** Shows the file path "interfaces.R — languageserver", the current line "Ln 2, Col 1", and other system information like "Spaces: 2", "UTF-8", "LF", and "R".

```
R > interfaces.R > base interfaces
Randy Lai, 8 months ago | 3 authors (Pierre Formont and others)
1 # base interfaces -->
2     Pierre Formont, a year ago • add more langfeatures documentation and interfaces
3 #' A position in a text document
4 #
5 #' Position in a text document expressed as zero-based line and zero-based
6 #' character offset.
7 #
8 #' @param line an integer
9 #' @param character an integer
10 #' @keywords internal
11 position <- function(line, character) {
12
13     if (!is.numeric(line) | !is.numeric(character)) {
14         stop("`position` requires numeric arguments!")
15     }
16
17     structure(
18         list(
19             line = line,
20             character = character
21         ),
22         class = "position"
23     )
24 }
25
26 print.position <- function(x, start_char = "", ...) {
27     cat(start_char, "<Position> Line:", x$line, " | Character:", x$character)
28 }
```

Code sections

- R code sections
- R Markdown code sections

The screenshot shows the VS Code interface with several open files:

- EXPLORER** view: Shows the file tree with "OPEN EDITORS" containing "init.R", "plot.R", and "vscode-R-test.R". It also lists "VS CODE-R" and "tests.txt".
- OUTLINE** view: Shows an outline of the code structure, including sections like "section 1" and "section 2", and functions like "fun", "fun2", "fun3", "function1", and "fun4".
- init.R**:

```
68 fun3 <- function(x, y) {  
69   # test  
70   z <- x + y  
71   z + 1  
72   z  
73 }  
74  
75 # section 2 #####  
76 test3 <- 1  
77 test4 <- 1  
78  
79 #' Hello  
80 #' World  
81 #' @param function1 new function  
82 function1 <- function(function1 = 1) {  
83   | function1  
84 }  
85  
86 #' A *new* function  
87 #' * This function does something  
88 #' * This is new  
89 fun4 <- function(x, y) {  
90   | # test  
91   | z <- x + y  
92   | z + 1  
93   | z  
94 }  
95
```
- plot.R**:

```
68 fun3 <- function(x, y) {  
69   # test  
70   z <- x + y  
71   z + 1  
72   z  
73 }  
74  
75 # section 2 #####  
76 test3 <- 1  
77 test4 <- 1  
78  
79 #' Hello  
80 #' World  
81 #' @param function1 new function  
82 function1 <- function(function1 = 1) {  
83   | function1  
84 }  
85  
86 #' A *new* function  
87 #' * This function does something  
88 #' * This is new  
89 fun4 <- function(x, y) {  
90   | # test  
91   | z <- x + y  
92   | z + 1  
93   | z  
94 }  
95
```
- vscode-R-test.R**:

```
68 fun3 <- function(x, y) {  
69   # test  
70   z <- x + y  
71   z + 1  
72   z  
73 }  
74  
75 # section 2 #####  
76 test3 <- 1  
77 test4 <- 1  
78  
79 #' Hello  
80 #' World  
81 #' @param function1 new function  
82 function1 <- function(function1 = 1) {  
83   | function1  
84 }  
85  
86 #' A *new* function  
87 #' * This function does something  
88 #' * This is new  
89 fun4 <- function(x, y) {  
90   | # test  
91   | z <- x + y  
92   | z + 1  
93   | z  
94 }  
95
```

Folding ranges

- R folding ranges
- R Markdown folding ranges

The screenshot shows the VS Code interface with a dark theme. The left sidebar has icons for Explorer, Open Editors, VS Code-R, Outline, and Timeline. The 'OPEN EDITORS' section lists 'test-doc.Rmd' and 'VS CODE-R'. The 'VS CODE-R' section lists 'test.R', 'test script.R', 'test-doc.Rmd', 'test0.R', and 'test1.R'. The 'OUTLINE' section shows a hierarchical tree of code blocks: 'hello' (containing 'subsection 1' with 'unnamed-chunk-1' and 'fun1', and 'subsection 2' with 'chunk-1' and 'fun2'). The main editor area displays the content of 'test-doc.Rmd' with folding markers (dashed lines) corresponding to the outline structure. The code includes R code and comments.

```
test-doc.Rmd — vscode-r
test-doc.Rmd ×
test-doc.Rmd > abc hello > abc subsection 1 > abc unnamed-chunk-1
1 --- ...
2 title: new document
3 ...
4 ...
5 # hello
6
7 Hello, world!
8
9 ## subsection 1
10
11 ...
12 x <- rnorm(100)
13 y <- rnorm(100)
14 m <- lm(y ~ x)
15
16 fun1 <- function(x, y) {
17   x + y
18   z <- local({
19     m <- x * y
20     n <- x - y
21     m + n
22   })
23 }
24 ...
```

R: (not attached) Ln 3, Col 4 Spaces: 2 UTF-8 LF R Markdown

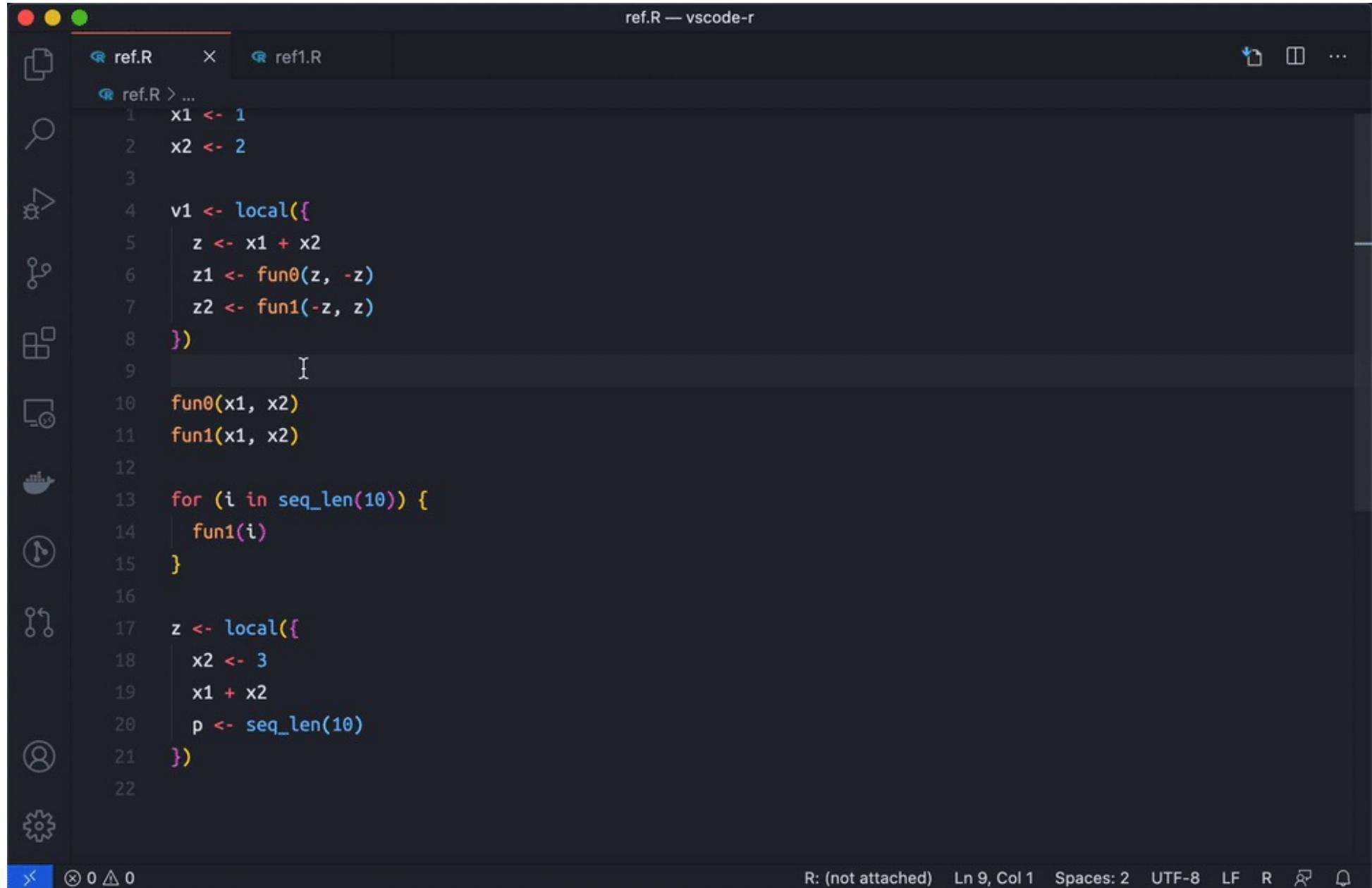
Find references

- Based on definition
- Project-wide searching
- Search all symbols that go to same definition

```
R > references.R > references_reply
You, a few seconds ago | 1 author (You)
1 references_xpath <- "//*[self::SYMBOL or self::SYMBOL_FUNCTION_CALL or self::SYMBOL_FORMALS]
2
3 #' @keywords internal
4 references_reply <- function(id, uri, workspace, document, point) {
5
6   token <- document$detect_token(point)
7   defn <- definition_reply(NULL, uri, workspace, document, point)
8   token_quote <- xml_single_quote(token$token)
9   You, 2 days ago • Add initial referencesProvider based on definition
10  logger$info("references_reply: ", list(
11    uri = uri,
12    token = token,
13    defn = defn$result
14  ))
15
16  result <- list()
17
18  if (length(defn$result)) {
19    for (doc_uri in workspace$documents$keys()) {
```

Rename symbol

- Based on references
- Project-wide rename



```
ref.R -- vscode-r
ref.R      ref1.R
ref.R > ...
1 x1 <- 1
2 x2 <- 2
3
4 v1 <- local({
5   z <- x1 + x2
6   z1 <- fun0(z, -z)
7   z2 <- fun1(-z, z)
8 })
9
10 fun0(x1, x2)
11 fun1(x1, x2)
12
13 for (i in seq_len(10)) {
14   fun1(i)
15 }
16
17 z <- local({
18   x2 <- 3
19   x1 + x2
20   p <- seq_len(10)
21 })
22
```

R: (not attached) Ln 9, Col 1 Spaces: 2 UTF-8 LF R ⚙️ 🔍 📡

Call Hierarchy

- Based on references
- Project-wide hierarchy
- Incoming calls
- Outgoing calls

The screenshot shows the RStudio IDE interface with the 'languageserver' extension active. The left sidebar displays the project structure in the Explorer view, including files like 'utils.R', 'language.R', 'selection.R', 'session.R', 'signature.R', 'symbol.R', 'task.R', 'utils.R', 'workspace.R', 'src', 'tests', 'testthat', 'helper-utils.R', and 'test-call-hierarchy.R'. The 'OUTLINE' section lists various utility functions such as 'capture_print', 'check_scope', 'code_point_from_unit', 'code_point_to_unit', 'convert_comment_to_documentation...', 'convert_doc_string', 'convert_doc_to_markdown', 'empty_string_to_null', 'equal_definition', 'equal_position', and '...'. The main editor window shows the 'utils.R' file with code related to call hierarchy and R Markdown files. A cursor is visible over the word 'uri' in the 'is_rmarkdown' function. The status bar at the bottom provides information about the current file and session.

```
utils.R — languageserver
R > utils.R > ...
146 equal_definition <- function(x, y) {
147   x$uri == y$uri && equal_range(x$range, y$range)
148 }
149 }
150 You, 2 days ago • Update call_hierarchy functions
151 #' Check if a file is an RMarkdown file
152 #' @keywords internal
153 is_rmarkdown <- function(uri) {
154   filename <- path_from_uri(uri)
155   endsWith(tolower(filename), ".rmd") || endsWith(tolower(filename), ".rmarkdown")
156 }
157
158 #' Check if a token is in a R code block in an Rmarkdown file
159 #
160 #' In an RMarkdown document, tokens can be either inside an R code block or
161 #' in the text. This function will return `FALSE` if the token is in the text
162 #' and `TRUE` if it is in a code block. For any other files, it always returns `TRUE`
163 #
164 #' @keywords internal
165 check_scope <- function(uri, document, point) {
166   if (is_rmarkdown(uri)) {
167     row <- point$row
168     flags <- vapply(
169       document$content[1:(row + 1)], startsWith, logical(1), "", USE.NAMES
170     )
171     if (any(flags)) {
172       last_match <- document$content[max(which(flags))]
173     }
174   }
175 }
```

You, 2 days ago R: (not attached) Ln 150, Col 1 Spaces: 4 UTF-8 LF R Git Graph

Selection range

- Based on parsed expression ranges

diagnostics.R — languageserver

EXPLORER

OPEN EDITORS

- helper-utils.R tests/testthat
- test-selection.R tests/testthat
- document.R R
- diagnostics.R R

LANGUAGESERVER

- man-roxygen
- R
 - capabilities.R
 - color.R
 - completion.R
 - definition.R
 - diagnostics.R
 - document.R
 - folding.R

OUTLINE

- diagnose_file
- diagnostic_from_lint
- diagnostic_range
- diagnostic_severity
- diagnostics_callback
- diagnostics_task
- find_config
- DiagnosticSeverity

diagnostics.R

```
115     )
116 }
117
118
119 diagnostics_task <- function(self, uri, document, delay = 0) {
120   version <- document$version
121   content <- document$content
122   create_task(
123     target = package_call(diagnose_file),
124     args = list(uri = uri, content = content),
125     callback = function(result) diagnostics_callback(self, uri, version, result),
126     error = function(e) {
127       logger$info("diagnostics_task:", e)
128       diagnostics_callback(self, uri, version, list(list(
129         range = range(
130           start = position(0, 0),
131           end = position(0, 0) You, a year ago via PR #220 • Call diagnostics_
132         ),
133         severity = DiagnosticSeverity$Error,
134         source = "lintr",
135         message = paste0("Failed to run diagnostics: ", conditionMessage(e))
136       )))
137     },
138     delay = delay
139   )
140 }
141 }
```

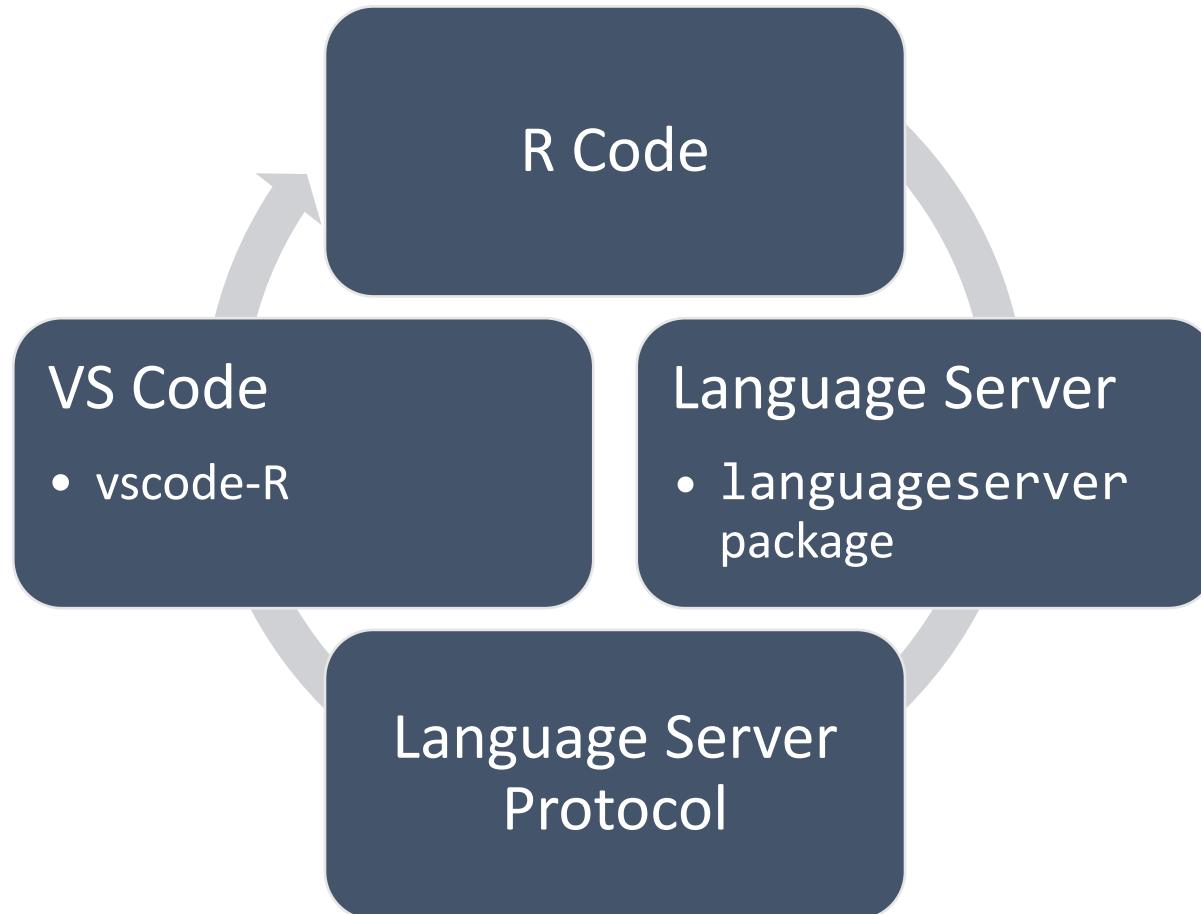
TIMELINE

NPM SCRIPTS

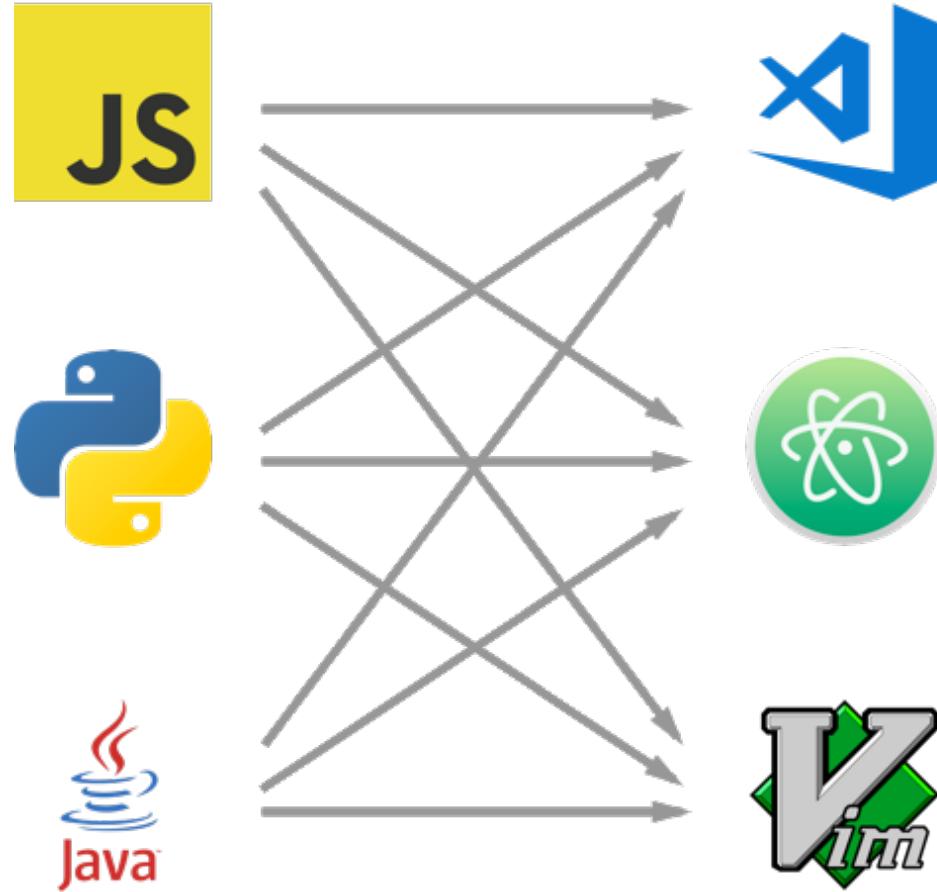
selection-range helper-utils.R test-selection.R document.R diagnostics.R

You, a year ago via PR #220 R: (not attached) Ln 131, Col 39 Spaces: 4 UTF-8 LF R

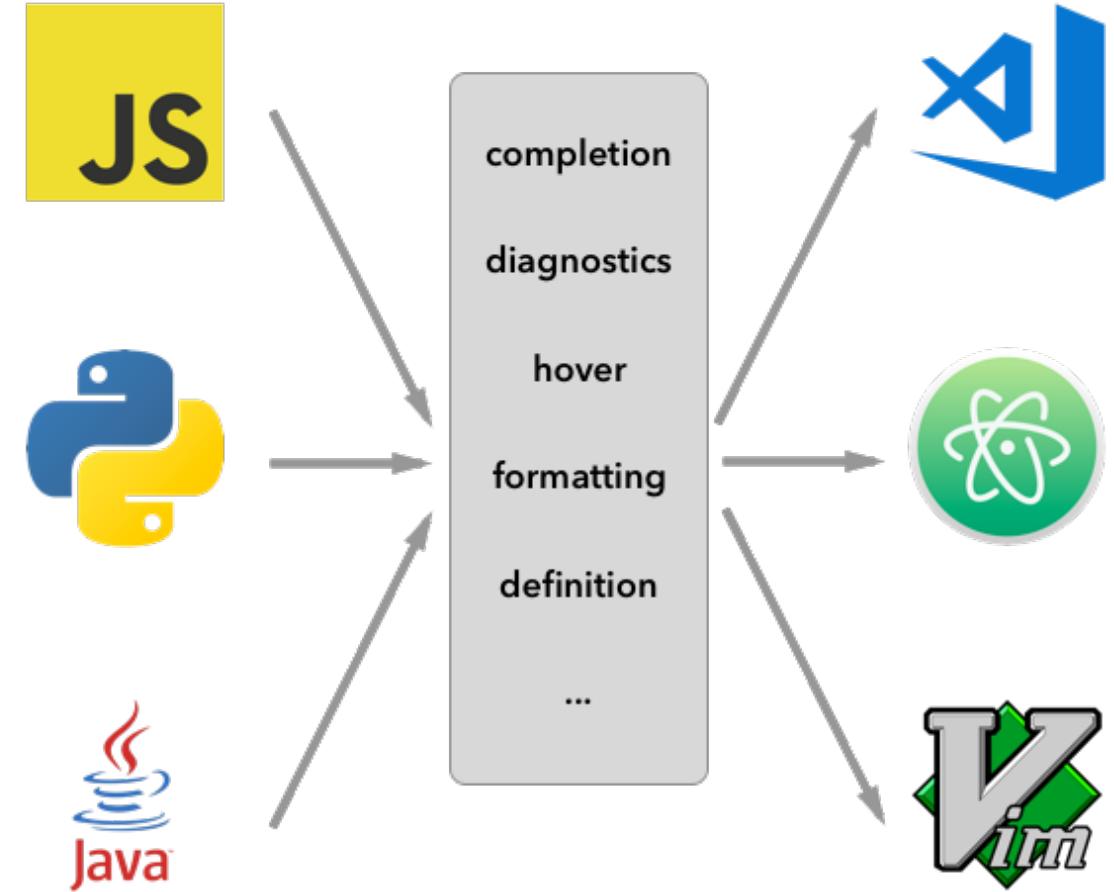
Behind the scene



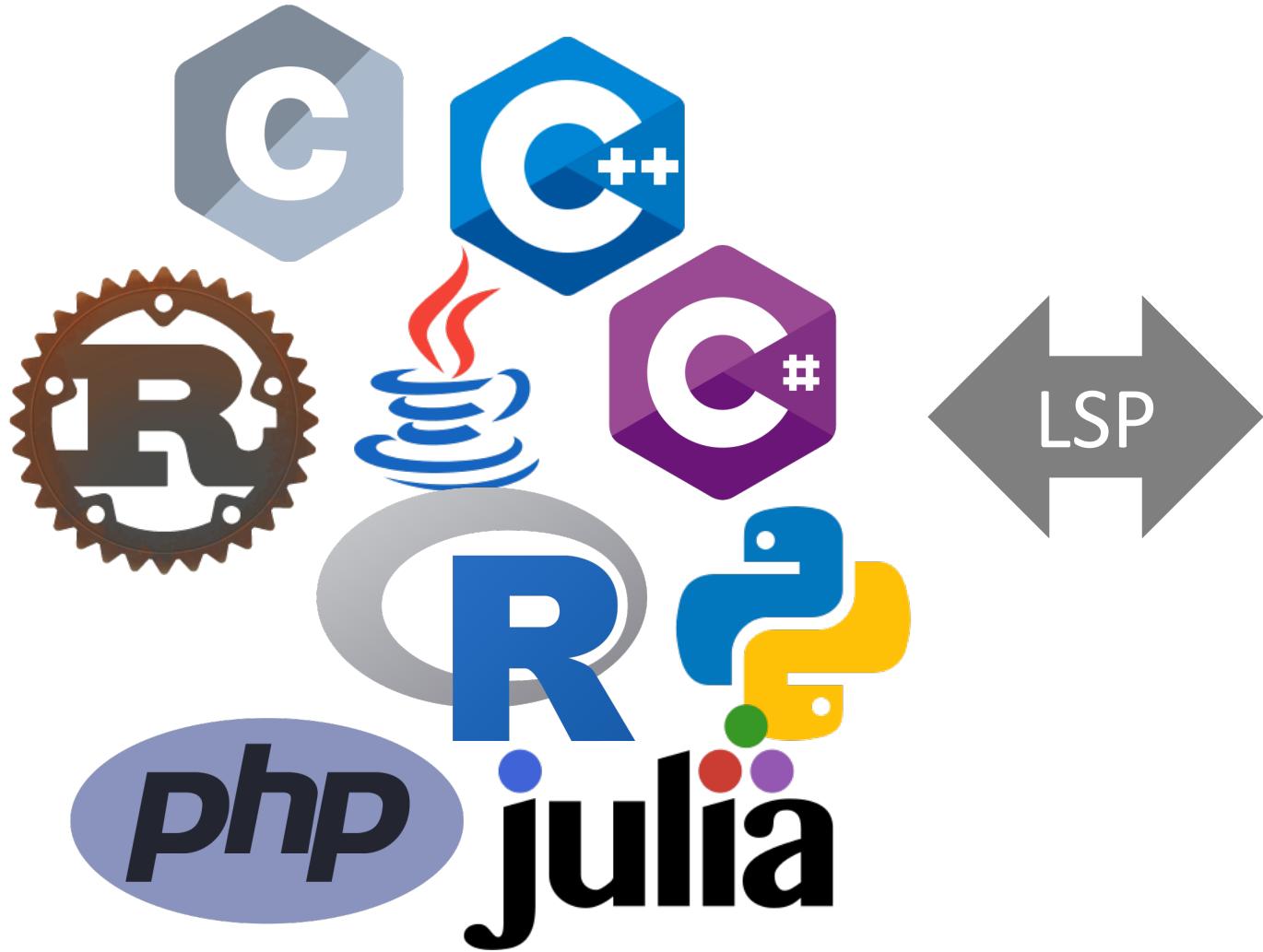
NO LSP



LSP



Language Server Protocol



R Language Server: the backend

textDocumentSync

hover

completion

signatureHelp

definition

Document
Highlight

Document
Symbol

Workspace
Symbol

Document
Formatting

Document
RangeFormatting

Document
OnTypeFormatting

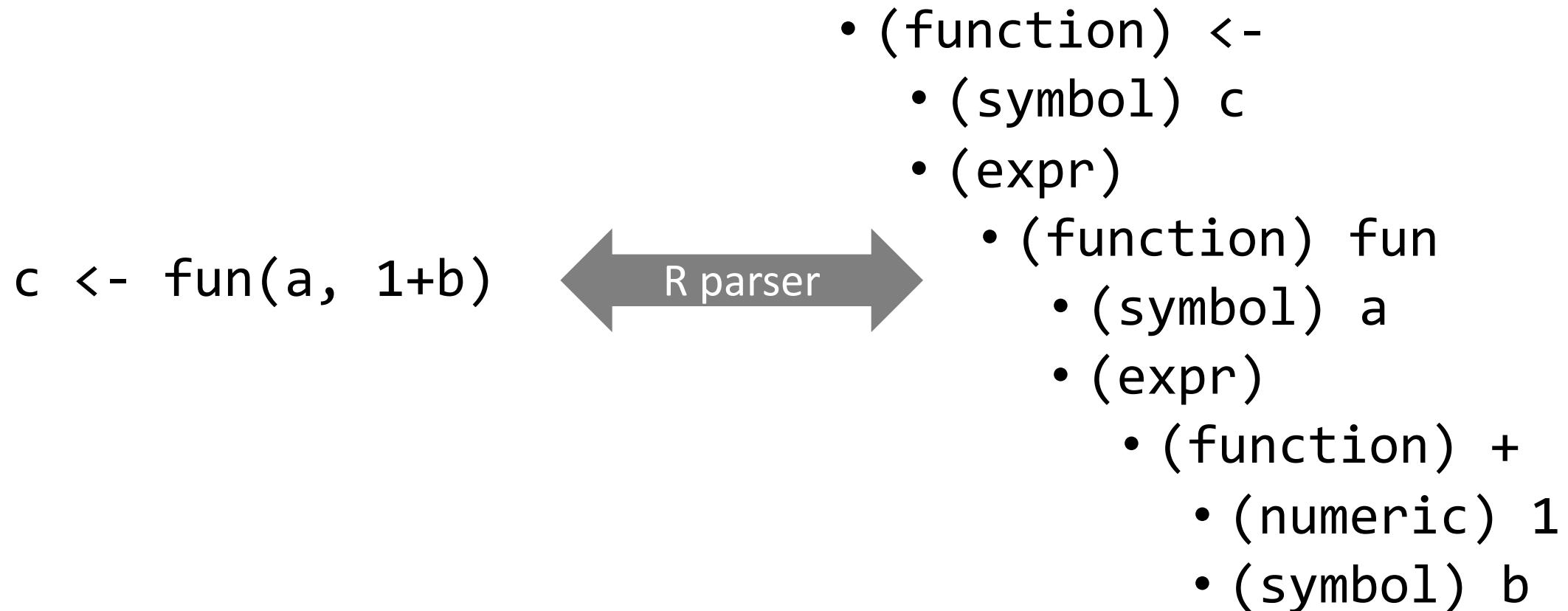
Document
Link

reference

rename

codeAction

Static code analysis based on Abstract-Syntax Tree (AST)



Why static/lexical analysis

- No need to execute user code
- Works in the background
- Works in parallel
- Still works if user interactive R session
 - Is not started
 - Is busy
 - Has crashed

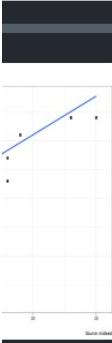
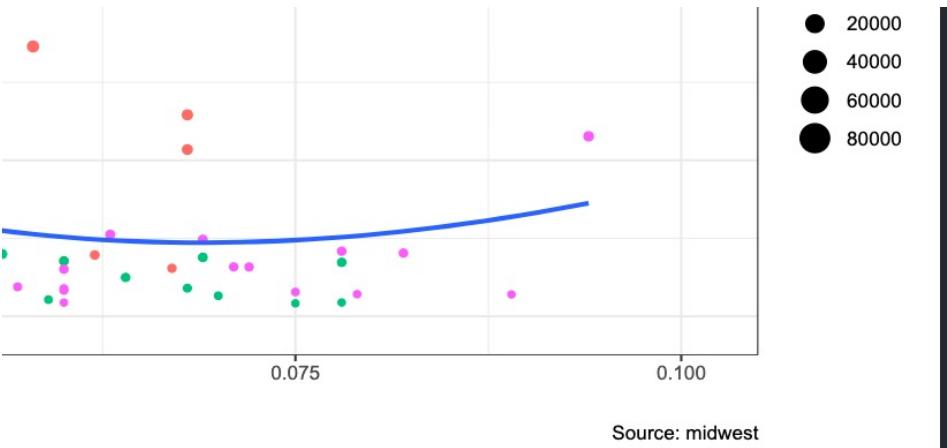
vscode-R extension: the frontend

- Communicate with R language server
- Provide useful commands
- Manage R sessions
- Send code to active terminal
- R Session Watcher
- R Help Viewer
- R Markdown support

vscode-R Session Watcher

- File-based vs. socket-based communication
- Session symbol hover
- Workspace viewer
- Data viewer: data frame, matrix, list
- Plot viewer: base graphics, ggplot2, lattice, etc.
- Browser: shiny apps, R Markdown server
- WebView: htmlwidgets, profvis, etc.
- RStudio addin support

Attach to active session



```
geom_smooth() using formula 'y ~ x'
Warning messages:
1: Removed 15 rows containing non-finite values (stat_smooth).
2: Removed 15 rows containing missing values (geom_point).

> # load package and data
library(ggplot2)

> data(mpg, package = "ggplot2") # alternate source: "http://goo.gl/uEeRGu")

> theme_set(theme_bw()) # pre-set the bw theme.

> g <- ggplot(mpg, aes(cty, hwy))

> # Scatterplot
g + geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(subtitle = "mpg: city vs highway mileage",
       y = "hwy",
       x = "cty",
       title = "Scatterplot with overlapping points",
       caption = "Source: midwest")

`geom_smooth()` using formula 'y ~ x'

> []
```

R: 76532 Ln 8, Col 1 Spaces: 2 UTF-8 LF R ⚙️ 🔔

Workspace viewer

- Global symbols
- Attached session

The screenshot shows the RStudio workspace viewer interface. At the top, there's a header with the letter 'R' and three icons: a folder, a square, and a document. To the right of the icons is a three-dot menu. Below the header, a section titled 'WORKSPACE' is expanded, showing a list of objects:

- > g List of 9
- > gg List of 9
- > midwest tbl_df: 437 obs. of 28 variables
- > midwest_select tbl_df: 6 obs. of 28 variables
- > mpg tbl_df: 234 obs. of 11 variables X ↻

The 'mpg' object is selected, and its details are shown in a expanded view below it:

- \$ manufacturer: chr [1:234] "audi" ...
- \$ model : chr [1:234] "a4" ...
- \$ displ : num [1:234] 1.8 1.8 ...
- \$ year : int [1:234] 1999 1999 ...
- \$ cyl : int [1:234] 4 4 ...
- \$ trans : chr [1:234] "auto(l5)" ...
- \$ drv : chr [1:234] "f" ...
- \$ cty : int [1:234] 18 21 ...
- \$ hwy : int [1:234] 29 29 ...
- \$ fl : chr [1:234] "p" ...
- \$ class : chr [1:234] "compact" ...

Hover: Symbol value in active session

The screenshot shows a Jupyter Notebook interface with two code cells and a terminal window.

Code Cell 1 (R-test1):

```
x <- rnorm(obs, mean = 1, sd = 1)
num [1:100] 2.4587 2.1123 1.4571 0.7022 0.0514 ...
x <- rnorm(obs, mean = 1, sd = 1)
y <- 2 * x + rnorm(obs)
w <- runif(obs)
m <- lm(y ~ x, weights = w)
coef(m)
```

Code Cell 2 (R-test):

```
result <- local({
  x <- rnorm(obs)
  y <- rnorm(obs)
  (x + y) * w
})
```

Terminal:

```
> obs <- 100
> x <- rnorm(obs, mean = 1, sd = 1)
> y <- 2 * x + rnorm(obs)
> w <- runif(obs)
> m <- lm(y ~ x, weights = w)
> coef(m)
(Intercept)           x
-0.3927031    2.0776163
> 
```

Data viewer

- View()

(row)	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4	
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02	0	1	4	
Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4	
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02	0	0	3	
Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3	
Duster 360	14.3	8	360	245	3.21	3.57	15.84	0	0	3	
Merc 240D	24.4	4	146.7	62	3.69	3.19	20	1	0	4	
Merc 230	22.8	4	140.8	95	3.92	3.15	22.9	1	0	4	
Merc 280	19.2	6	167.6	123	3.92	3.44	18.3	1	0	4	
Merc 280C	17.8	6	167.6	123	3.92	3.44	18.9	1	0	4	
Merc 450SE	16.4	8	275.8	180	3.07	4.07	17.4	0	0	3	
Merc 450SL	17.3	8	275.8	180	3.07	3.73	17.6	0	0	3	
Merc 450SLC	15.2	8	275.8	180	3.07	3.78	18	0	0	3	
Cadillac Fleetwood	10.4	8	472	205	2.93	5.25	17.98	0	0	3	
Lincoln Continental	10.4	8	460	215	3	5.424	17.82	0	0	3	
Chrysler Imperial	14.7	8	440	230	3.23	5.345	17.42	0	0	3	
Fiat 128	32.4	4	78.7	66	4.08	2.2	19.47	1	1	4	
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.9	1	1	4	
Toyota Corona	21.5	4	120.1	97	3.7	2.465	20.01	1	0	3	
Dodge Challenger	15.5	8	318	150	2.76	3.52	16.87	0	0	3	
AMC Javelin	15.2	8	304	150	3.15	3.435	17.3	0	0	3	
Camaro Z28	13.3	8	350	245	3.73	3.84	15.41	0	0	3	
Pontiac Firebird	19.2	8	400	175	3.08	3.845	17.05	0	0	3	
Fiat X1-9	27.3	4	79	66	4.08	1.935	18.9	1	1	4	
Porsche 914-2	26	4	120.3	91	4.43	2.14	16.7	0	1	5	

List viewer

```
mtcars jsonlite::read_json("package.json") ×

{

  "name": "r",
  "displayName": "R",
  "description": "R Extension for Visual Studio Code",
  "version": "2.1.0",
  "author": "REditorSupport",
  "license": "SEE LICENSE IN LICENSE",
  "publisher": "Ikuyadeu",
  "icon": "images/Rlogo.png",
  "repository": {
    "type": "git",
    "url": "https://github.com/REditorSupport/vscode-R"
  },
  "bugs": {
    "url": "https://github.com/REditorSupport/vscode-R/issues"
  },
  "categories": [
    "Programming Languages",
    "Snippets",
    "Other"
  ],
  "keywords": [
    "R",
    "R language",
    "R documentation",
    "R Markdown"
  ],
  "engines": {
    "vscode": "^1.57.0"
  },
  "activationEvents": [
    "onLanguage:r",
    "onLanguage:rd",
    "onLanguage:rmd",
    "onLanguage:debian-control.r",
    "workspaceContains:*.{rproj,Rproj,r,R,rd,Rd,rmd,Rmd}",
    "onCommand:r.createRTerm",
    "onCommand:r.runSource",
    "onCommand:r.knitRmd",
    "onCommand:r.knitRmdToPdf",
    "onCommand:r.knitRmdToHtml",
    "onCommand:r.knitRmdToAll",
    "onCommand:r.runSourcewithEcho",
    "onCommand:r.runSelection",
    "onCommand:r.runSelectionInActiveTerm",
    "onCommand:r.selectCurrentChunk"
  ]
}
```

Plot viewer: PNG viewer

The screenshot shows the VS Code interface with the following components:

- Left Panel:** A sidebar with various icons for file operations like Open, Save, Find, and Settings.
- Code Editor:** An R script titled "vscode-R-test.R" containing the following code:

```
19
20   plot(rnorm(100))
21   abline(h = 0, col = "blue")
22
23   library(ggplot2)
24   ggplot(mpg, aes(displ, hwy, colour = class)) +
25     geom_point()
26
27   library(plotly)
28   p <- ggplot(data = diamonds, aes(x = cut,
29               geom_bar(position = "dodge"))
30   ggplotly(p)
31
32   library(magrittr)
33   library(highcharter)
34   highchart() %>%
35     hc_title(text = "Scatter chart with si")
36     hc_add_series_scatter(mtcars$wt, mtcars
37       mtcars$drat, mtcars$hp)
38
39   library(visNetwork)
```
- Preview View:** A preview of a scatter plot titled "plot.png". The plot shows "hwy" on the y-axis (ranging from 20 to 40) versus "displ" on the x-axis (ranging from 2 to 7). Data points are colored by "class", with a legend on the right side. The legend includes: 2seater (red), compact (orange), midsize (green), minivan (teal), pickup (blue), subcompact (purple), and suv (pink).
- Terminal View:** A terminal window titled "vscode-R-test.R — vscode-r" showing R session history:

```
> res <- lapply(1:10, rnorm)

> test1 <- rnorm(100)

> test2 <- function(x, y) {
+   x + y
}

> test2(1, 2)
[1] 3

> data1 <- list(a = 1, b = 2)

> plot(rnorm(100))

> abline(h = 0, col = "blue")

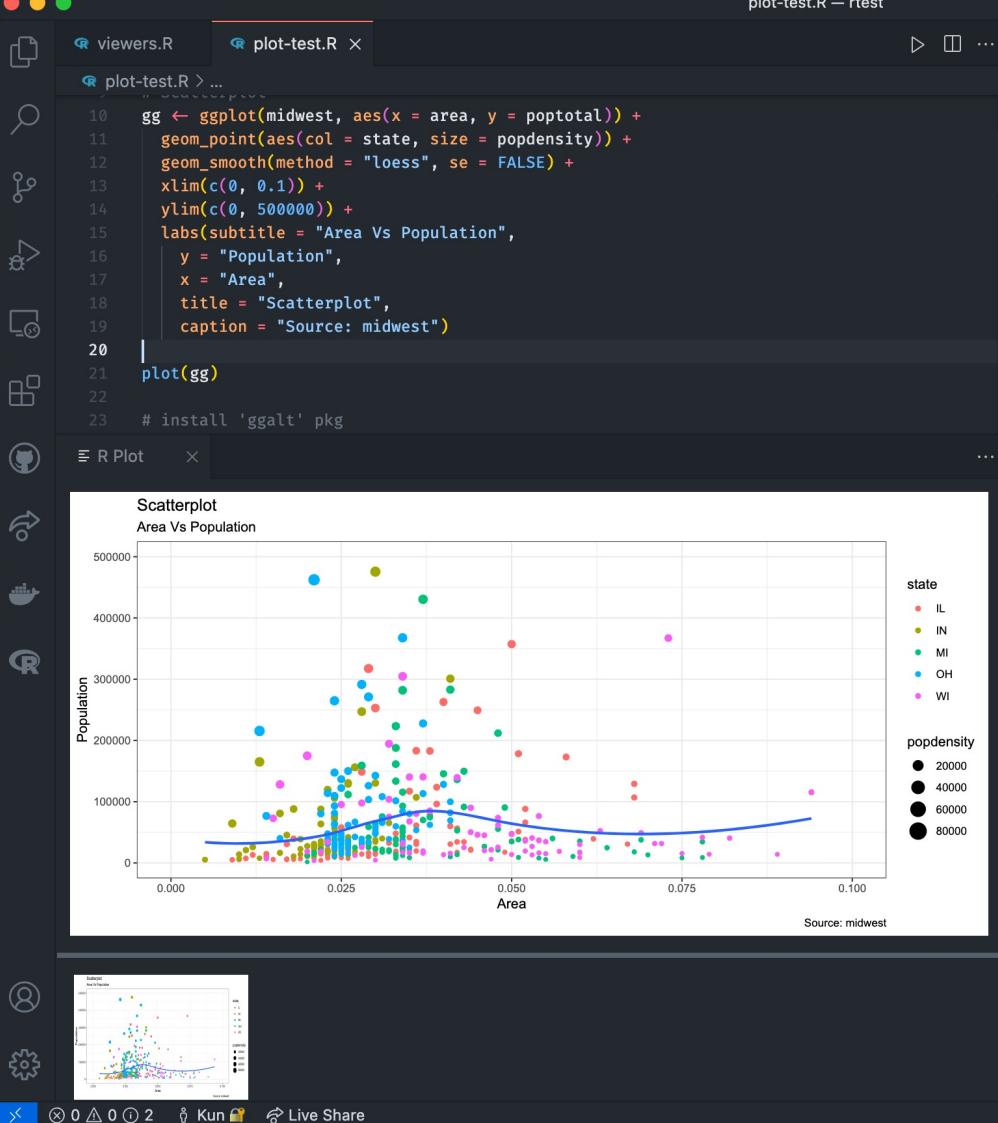
> library(ggplot2)

> ggplot(mpg, aes(displ, hwy, colour = class)) +
  geom_point()

> 
```
- Bottom Status Bar:** Shows the current R session ID (R: 78684), line number (Ln 25), column number (Col 15), spaces used (Spaces: 2), encoding (UTF-8), line feed (LF), and other status indicators.

Plot viewer

- Based on `http://pgd`
- SVG rendering
- Plot history
- Auto resize
- Open in browser
- Save to multiple formats



The screenshot shows the RStudio interface with the following components:

- Code Editor:** The top-left pane displays the R script `plot-test.R` containing the following code:

```
10 gg <- ggplot(midwest, aes(x = area, y = poptotal)) +  
11   geom_point(aes(col = state, size = popdensity)) +  
12   geom_smooth(method = "loess", se = FALSE) +  
13   xlim(c(0, 0.1)) +  
14   ylim(c(0, 500000)) +  
15   labs(subtitle = "Area Vs Population",  
16        y = "Population",  
17        x = "Area",  
18        title = "Scatterplot",  
19        caption = "Source: midwest")  
20  
21 plot(gg)  
22  
23 # install 'ggalt' pkg
```
- Console:** The right-hand panel shows the R session output:

```
> rtest  
R version 4.1.0 (2021-05-18) -- "Camp Pontanezen"  
Platform: x86_64-apple-darwin17.0 (64-bit)  
> View(mtcars)  
> # install.packages("ggplot2")  
# load package and data  
options(scipen = 999) # turn-off scientific notation like 1e+48  
library(ggplot2)  
theme_set(theme_bw()) # pre-set the bw theme.  
data("midwest", package = "ggplot2")  
# midwest <- read.csv("http://go.gli/G1K1K") # bkup data source  
  
# Scatterplot  
gg <- ggplot(midwest, aes(x = area, y = poptotal)) +  
  geom_point(aes(col = state, size = popdensity)) +  
  geom_smooth(method = "loess", se = FALSE) +  
  xlim(c(0, 0.1)) +  
  ylim(c(0, 500000)) +  
  labs(subtitle = "Area Vs Population",  
       y = "Population",  
       x = "Area",  
       title = "Scatterplot",  
       caption = "Source: midwest")  
  
plot(gg)  
geom_smooth() using formula 'y ~ x'  
Warning messages:  
1: Removed 15 rows containing non-finite values (stat_smooth).  
2: Removed 15 rows containing missing values (geom_point).  
  
> .vsc.attach()  
> []
```
- Plot Area:** The bottom-right pane displays the generated scatterplot titled "Scatterplot" with the subtitle "Area Vs Population". The x-axis is labeled "Area" and ranges from 0.00 to 0.10. The y-axis is labeled "Population" and ranges from 0 to 500,000. Data points are colored by state (IL, IN, MI, OH, WI) and sized by population density (20000, 40000, 60000, 80000). A blue loess smoothing line is overlaid on the plot.
- Bottom Status Bar:** The status bar at the bottom provides information about the R session: "R: (not attached) Ln 20, Col 1 Spaces: 2 UTF-8 LF R".

Show htmlwidgets in WebView

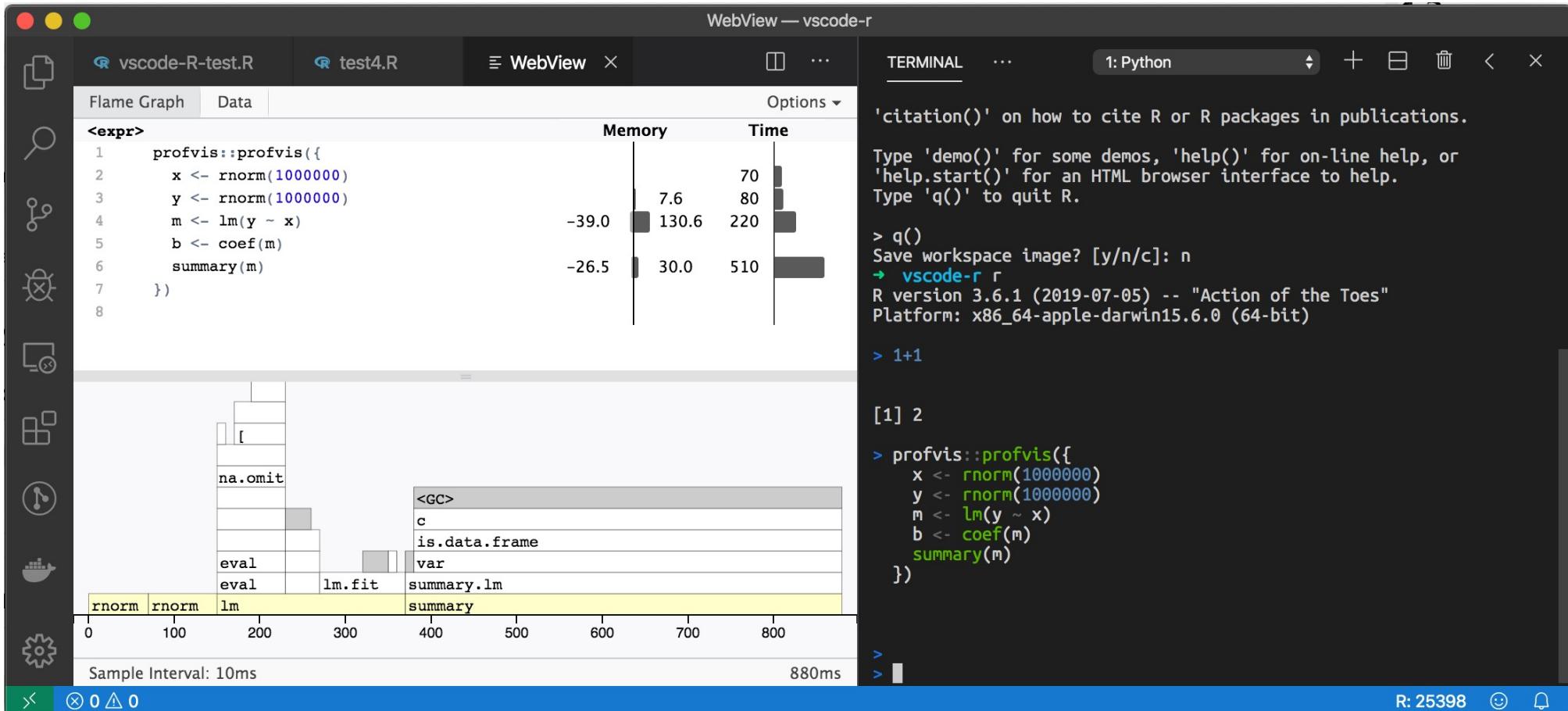
The screenshot shows the Visual Studio Code interface with the following components:

- Left Sidebar:** Includes icons for file operations, search, symbols, status, and settings.
- Editor Area:** A tab labeled "vscode-R-test.R" contains R code:

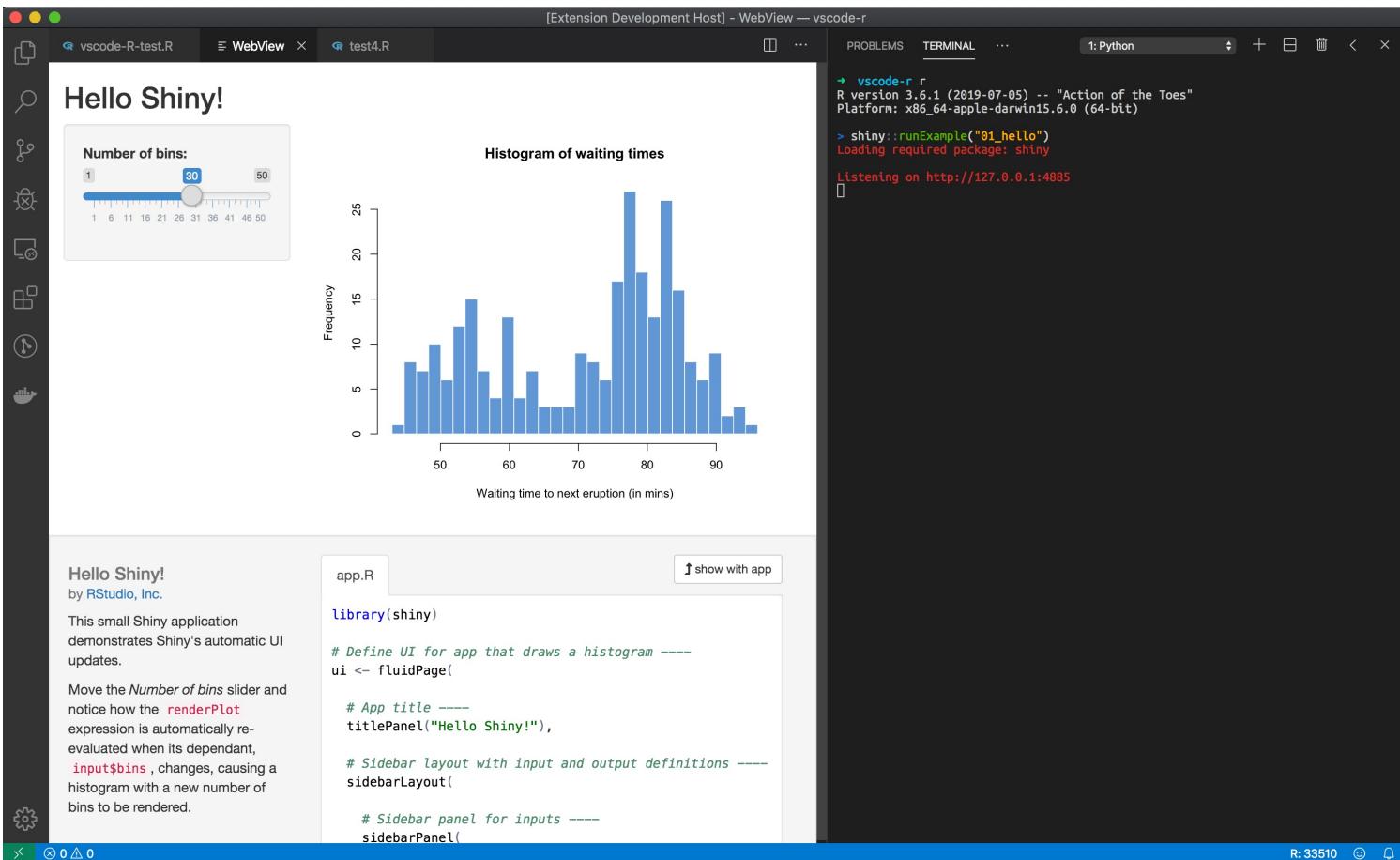
```
34 highchart() %>%
35   hc_title(text = "Scatter chart with size and color") %>%
36   hc_add_series_scatter(mtcars$wt, mtcars$mpg,
37   mtcars$drat, mtcars$hp)
38
39 library(visNetwork)
```
- Preview Area:** Below the editor, there are three tabs: "plot.png" (disabled), "WebView" (selected), and another "WebView". The "WebView" tab displays a scatter plot titled "Scatter chart with size and color". The plot shows the relationship between car weight (wt) on the x-axis and fuel efficiency (mpg) on the y-axis. Data points are colored by engine displacement (drat) and have varying sizes based on horsepower (hp). The plot area includes horizontal grid lines at 0, 10, 20, 30, and 40.
- Terminal Area:** Labeled "TERMINAL" and "1: Python", it shows R session output:

```
> p <- ggplot(data = diamonds, aes(x = cut, fill =
  clarity)) +
  geom_bar(position = "dodge")
> ggplotly(p)
> library(magrittr)
> library(highcharter)
Registered S3 method overwritten by 'xts':
  method      from
  as.zoo.xts zoo
Registered S3 method overwritten by 'quantmod':
  method      from
  as.zoo.data.frame zoo
Highcharts (www.highcharts.com) is a Highsoft software product which is
not free for commercial and Governmental use
> highchart() %>%
  hc_title(text = "Scatter chart with size and c
  olor") %>%
  hc_add_series_scatter(mtcars$wt, mtcars$mpg,
  mtcars$drat, mtcars$hp)
Warning message:
'hc_add_series_scatter' is deprecated.
Use 'hc_add_series' instead.
See help("Deprecated")
> 
```
- Bottom Status Bar:** Shows "R: 78684 Ln 38, Col 1 Spaces: 2 UTF-8 LF R ☺ 📲".

Code profiling by profvis

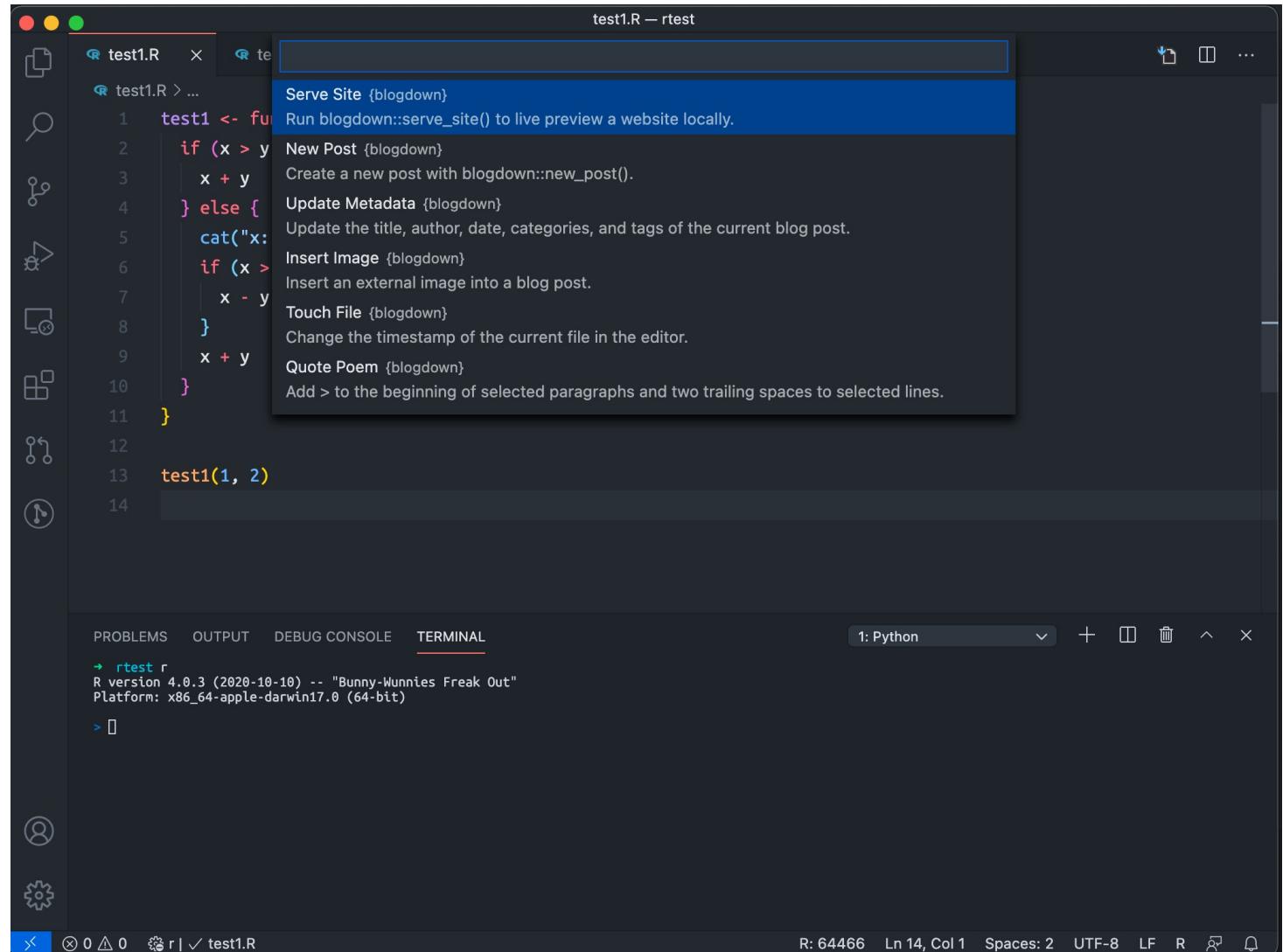


Show Shiny apps



RStudio addin support

- {rstudioapi} emulation layer
- Support many addins



Help viewer

The screenshot shows the RStudio interface with the Help viewer open. The title bar says "R Help — rtest". The left sidebar has icons for files, search, and other tools. The main area shows the R code for creating a scatterplot:

```
plot-test.R > ...
1 # install.packages("ggplot2")
2 # load package and data
3 options(scipen = 999) # turn-off scientific notation like 1e+48
4 library(ggplot2)
5 theme_set(theme_bw()) # pre-set the bw theme.
6 data("midwest", package = "ggplot2")
7 # midwest <- read.csv("http://goo.gl/G1K41K") # bkup data source
8
9 # Scatterplot
10 gg <- ggplot(midwest, aes(x = area, y = poptotal)) +
11   geom_point(aes(col = state, size = popdensity)) +
12   geom_smooth(method = "loess", se = FALSE) +
13   xlim(c(0, 0.1)) +
14   ylim(c(0, 500000)) +
15   labs(subtitle = "Area Vs Population",
16        y = "Population",
17        x = "Area",
18        title = "Scatterplot",
19        caption = "Source: midwest")
20
```

The Help viewer window has a title "Create a new ggplot". It contains sections for "Description" and "Usage".

Description

ggplot() initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

Usage

```
ggplot(data = NULL, mapping = aes(), ... , environment = parent.frame())
```

Arguments

data Default dataset to use for plot. If not already a data.frame, will be converted to one by [fortify\(\)](#). If not specified, must be supplied in each layer added to the plot.

mapping Default list of aesthetic mappings to use for plot. If not specified, must be supplied in each layer added to the plot.

... Other arguments passed on to methods. Not currently used

The right side of the screen shows the R console output:

```
PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL Python +
> ?plotly::ploty
No documentation for 'ploty' in specified packages and libraries:
you could try ??ploty'
> ?plotly::plotly
> library(plotly)
Attaching package: 'plotly'
The following object is masked from 'package:ggplot2':
  last_plot
The following object is masked from 'package:stats':
  filter
The following object is masked from 'package:graphics':
  layout

> ggplot(penguins, aes(bill_length_mm, body_mass_g,
  data = palmerpenguins::penguins, color = species))
Error in loadNamespace(x) : there is no package called 'palmerpenguins'
Backtrace:
1: stop(cond)
2: doWithOneRestart(return(expr), restart)
3: withOneRestart(expr, restarts[[1L]])
4: withRestarts(stop(cond), retry_loadNamespace = function() NULL)
5: loadNamespace(x)
6: ggplot(data, mapping, environment = caller_env)
7: qplot(bill_length_mm, body_mass_g, data = palmerpenguins::penguins,
  > data(canada.cities, package = "maps")
> viz <- ggplot(canada.cities, aes(long, lat)) +
  borders(regions = "canada") +
  coord_equal() +
  geom_point(aes(text = name, size = pop), colour = "red", alpha = 1 / 2)
Warning message:
Ignoring unknown aesthetics: text
> ggplotly(viz, tooltip = c("text", "size"))
> shiny::runExample("01_hello")
Loading required package: shiny
Listening on http://127.0.0.1:3892
Browsing http://127.0.0.1:3892
^C^C^C^C^Z
[1] + 10054 suspended  radian
→ rtest kill -9 10054
→ rtest
[1] + 10054 killed    radian
→ rtest
R version 4.1.0 (2021-05-18) -- "Camp Pontanezen"
Platform: x86_64-apple-darwin17.0 (64-bit)
> library(ggplot2)
Keep up to date with changes at https://www.tidyverse.org/blog/
> ?ggplot
> []
```

R: 10665

R Markdown: Editing

The screenshot shows the Visual Studio Code interface with an R Markdown document open. The left sidebar contains the Explorer, Outline, and NPM Scripts sections. The main editor area displays the R Markdown code, which includes a title, two sections, and various R code chunks. The right side features a terminal window showing the output of an R session.

```
# Title
This is a test document.

## Section 1

Run Chunk | Run Above
```{r}
...
```

Run Chunk | Run Above
```{r chunk-1}
x <- rnorm(100)
y <- rnorm(100)
...
```

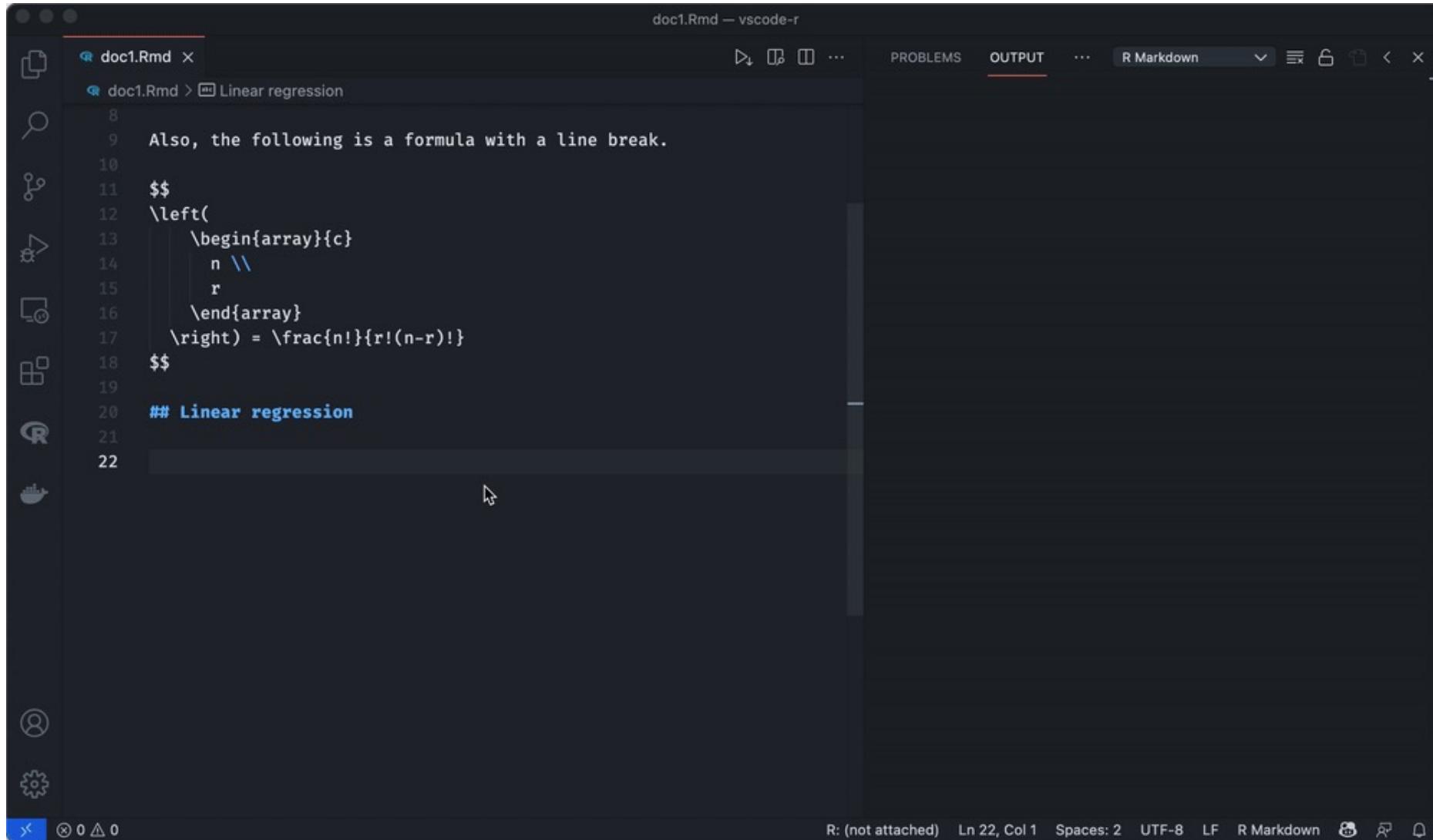
Run Chunk | Run Above
```{r plot1}
plot(rnorm(100))
abline(h = 0, col = "red")
...
```

## Section 2

# Terminal # Python
rtest r
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Platform: x86_64-apple-darwin17.0 (64-bit)
```

R: 69124 Ln 8, Col 1 Spaces: 2 UTF-8 LF R Markdown

R Markdown: Live preview



The screenshot shows a dark-themed instance of Visual Studio Code (VS Code) with a file named "doc1.Rmd" open. The code editor displays the following R Markdown code:

```
doc1.Rmd — vscode-r
doc1.Rmd ×
doc1.Rmd > Linear regression
8
9  Also, the following is a formula with a line break.
10
11 $$
12 \left(
13   \begin{array}{c}
14     n \\
15     r
16   \end{array}
17 \right) = \frac{n!}{r!(n-r)!}
18 $$
19
20 ## Linear regression
21
22
```

The code includes a mathematical formula involving factorials and binomial coefficients. The formula is split across multiple lines, demonstrating how line breaks are handled in R Markdown. The "Linear regression" section is also visible.

◀ HELP PAGES

- 🏠 Home
- ⚡ Open Help Topic using `?`
- 🔍 Search Help Topics using `??`
- abc Open Help Page for Selected Text
- ⟳ Clear Cached Index Files & Restart Hel...
- ⬇️ Install CRAN Package

◀ ⓢ Help Topics by Package

> anytime

> arrow



☰ Index

📄 DESCRIPTION

- 'match' and '%in%' for Arrow objects
- 'table' for Arrow objects
- Apache Arrow data types
- Apply a function to a stream of RecordB...

● ArrayData class

● Arrow Arrays

● Arrow CSV and JSON table reader class...

● Arrow expressions

● Arrow scalars

● Buffer class

● Call an Arrow compute function

● Check whether a compression codec is ...

● ChunkedArray class

● class arrow::DataType



Package management

tidyver

tidyverse

Easily Install and Load the '**Tidyverse**'

moderndive

Tidyverse-Friendly Introductory Linear Regression

omopr

OMOP CDM Databases using the **Tidyverse**

rlang

Functions for Base Types and Core R and '**Tidyverse**' Features

tidyboot

Tidyverse-Compatible Bootstrapping

tidycensus

Load US Census Boundary and Attribute Data as '**tidyverse**' and 'sf'-Ready Data Frames

tidyfst

Tidy Verbs for Fast Data Manipulation

tidyft

Tidy Verbs for Fast Data Operations by Reference

tidygenomics

Tidy Verbs for Dealing with Genomic Data Frames

Multi-session support

- Multiple VS Code terminals
- Running R sessions in tmux or screen windows
- Customizable layout with panes and windows
- Persistence with tmux/screen server

The screenshot shows the Visual Studio Code (VS Code) interface with a dark theme. In the center, there are two tabs for R scripts: "test.R" and "test2.R". The "test.R" tab is active, displaying the following R code:

```
1 x <- rnorm(100)
2 y <- rnorm(100)
3 m <- lm(y ~ x)
4 summary(m)
5
```

Below the tabs, the "TERMINAL" tab is selected, showing a tmux session titled "1: tmux". The terminal pane displays the command "vscode-r" followed by a red arrow icon. At the bottom of the terminal, there is a status bar with the text "0 0:zsh*".

The left side of the interface features a vertical sidebar with various icons for file operations like copy, paste, search, and refresh, as well as icons for Git, file navigation, and settings.

The bottom of the screen includes a status bar with performance metrics: "0.62% [] 0.1% | 2020-04-19 | 05:31:34 | ken@home-server". It also shows the current file path: "R: 28328 Ln 5, Col 1". The status bar also includes icons for file type (LF), encoding (UTF-8), and other settings.

VS Code R Debugger

- Debug R script
- Debug R package
- Set breakpoints
- Variables and environments
- Call stack
- Interactive debug console
- Inline viewer on hover

The screenshot shows the VS Code interface with the R Debugger extension installed. The main editor window displays an R script named 'debugger.R' with syntax highlighting. A cursor is positioned over the code. The left sidebar features several panels: 'VARIABLES', 'WATCH', and 'CALL STACK' under the 'BREAKPOINTS' section. The bottom left also shows a 'BREAKPOINTS' panel with two checkboxes: 'Errors from R file' (checked) and 'Errors from debug console'. The bottom navigation bar includes tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE' (which is selected), and 'TERMINAL'. A status bar at the bottom right shows file statistics: R: 81121 Ln 8, Col 8 Spaces: 2 UTF-8 LF R.

```
debugger.R — vscode-r
debugger.R X {} launch.json
debugger.R > which_cumsum_until
1  which_cumsum_until <- function(x, value) {
2    sum <- 0
3    for (i in seq_along(x)) {
4      if (!is.na(x[[i]])) {
5        sum <- sum + x[[i]]
6        if (sum > value) {
7          return(i)
8        }
9      }
10    }
11    return(NA_integer_)
12  }

14  test <- function(n, value, times) {
15    res <- vapply(seq_len(times), function(t) {
16      x <- rnorm(n)
17      i <- which_cumsum_until(x)
18      i
19    }, integer(1L))
20    quantile(res, seq(0, 1, 0.1), na.rm = TRUE)
21  }

23  qs <- test(100, 5, 100)
24  qs
25
```

More VS Code extensions

C/C++

Python

Jupiter

Output
Colorizer

TODO
Highlight

CMake

CMake Tools

Docker

Git Graph

Error Lens

GitLens

markdownlint

Path
Autocomplete

vscode-pdf

...

Real-time collaboration via Live Share

The screenshot shows two instances of Visual Studio Live Share, each displaying an R workspace. Both instances are running the same R script, `plot-test.R`, which generates a scatterplot titled "Scatterplot Area Vs Population".

Script Content:

```
# install.packages("ggplot2")
# load package and data
options(scipen = 999) # turn-off scientific notation like 1e+48
library(ggplot2)
theme_set(theme_bw()) # pre-set the bw theme.
data("midwest", package = "ggplot2")
# midwest <- read.csv("http://goo.gl/G1K41k") # bkp data source

# Scatterplot
gg <- ggplot(midwest, aes(x = area, y = pop))
geom_point(aes(col = state, size = popdensity))
geom_smooth(method = "loess", se = TRUE)
xlim(c(0, 0.1))
ylim(c(0, 500000))
labs(subtitle = "Area Vs Population",
y = "Population",
x = "Area",
title = "Scatterplot",
caption = "Source: midwest")

plot(gg)
```

Output:

Both instances show the same scatterplot with data points colored by state and sized by population density. A blue loess smoothing line is overlaid on the plot.

Terminal Output:

```
vtest R
R version 4.1.0 (2021-05-18) -- "Camp Pontanez"
Platform: x86_64-apple-darwin17.0 (64-bit)
> # install.packages("ggplot2")
> # load package and data
> options(scipen = 999) # turn-off scientific notation like 1e+48
> library(ggplot2)
> theme_set(theme_bw()) # pre-set the bw theme.

> data("midwest", package = "ggplot2")
> # midwest <- read.csv("http://goo.gl/G1K41k") # bkp data source

> # Scatterplot
> gg <- ggplot(midwest, aes(x = area, y = pop))
> geom_point(aes(col = state, size = popdensity))
> geom_smooth(method = "loess", se = TRUE)
> xlim(c(0, 0.1))
> ylim(c(0, 500000))
> labs(subtitle = "Area Vs Population",
> y = "Population",
> x = "Area",
> title = "Scatterplot",
> caption = "Source: midwest")

> plot(gg)
geom_smooth(): using formula 'y ~ x'
Warning messages:
1: Removed 35 rows containing non-finite values (stat_smooth).
2: Removed 15 rows containing missing values (geom_point).
```

C++ Development with Rcpp/cpp11

The screenshot shows a dark-themed instance of Visual Studio Code (VS Code) with the title "test.cpp — vscode-rcpp-demo [SSH: office]".

Explorer View: Shows the project structure with the following files and folders:

- OPEN EDITORS:
 - {} launch.json /home/renkun/W...
 - README.md /home/renkun/W...
 - X G test.cpp /home/renkun/... M
 - R debug.R /home/renkun/Works...
- VSCODE-RCPP-DEMO [SSH: OFFICE]:
 - {} c_cpp_properties.json
 - R debug.R
 - {} launch.json
 - {} tasks.json
 - > man
 - R
 - R package.R
 - R RcppExports.R
 - R test.R
 - src
 - ! .clang-format
 - .gitignore
 - G RcppExports.cpp
 - E RcppExports.o
 - G test.cpp M
 - E test.o
 - E VSCodeRcppDemo.so
 - tests
 - testthat
- > OUTLINE

Main Editor: Displays the content of `test.cpp`:

```
#include <Rcpp.h>
using namespace Rcpp;
double calc_sum(NumericVector x) {
    double sum = 0;
    for (int i = 0; i < x.size(); ++i)
```

Status Bar: Shows "SSH: office", "master*", and other developer information like file changes (0△0), Git Graph, and a timestamp "You, a day ago".

<https://github.com/renkun-ken/vscode-rcpp-demo>

C++ Debugging with Rcpp/cpp11

The screenshot shows the VS Code interface with the following details:

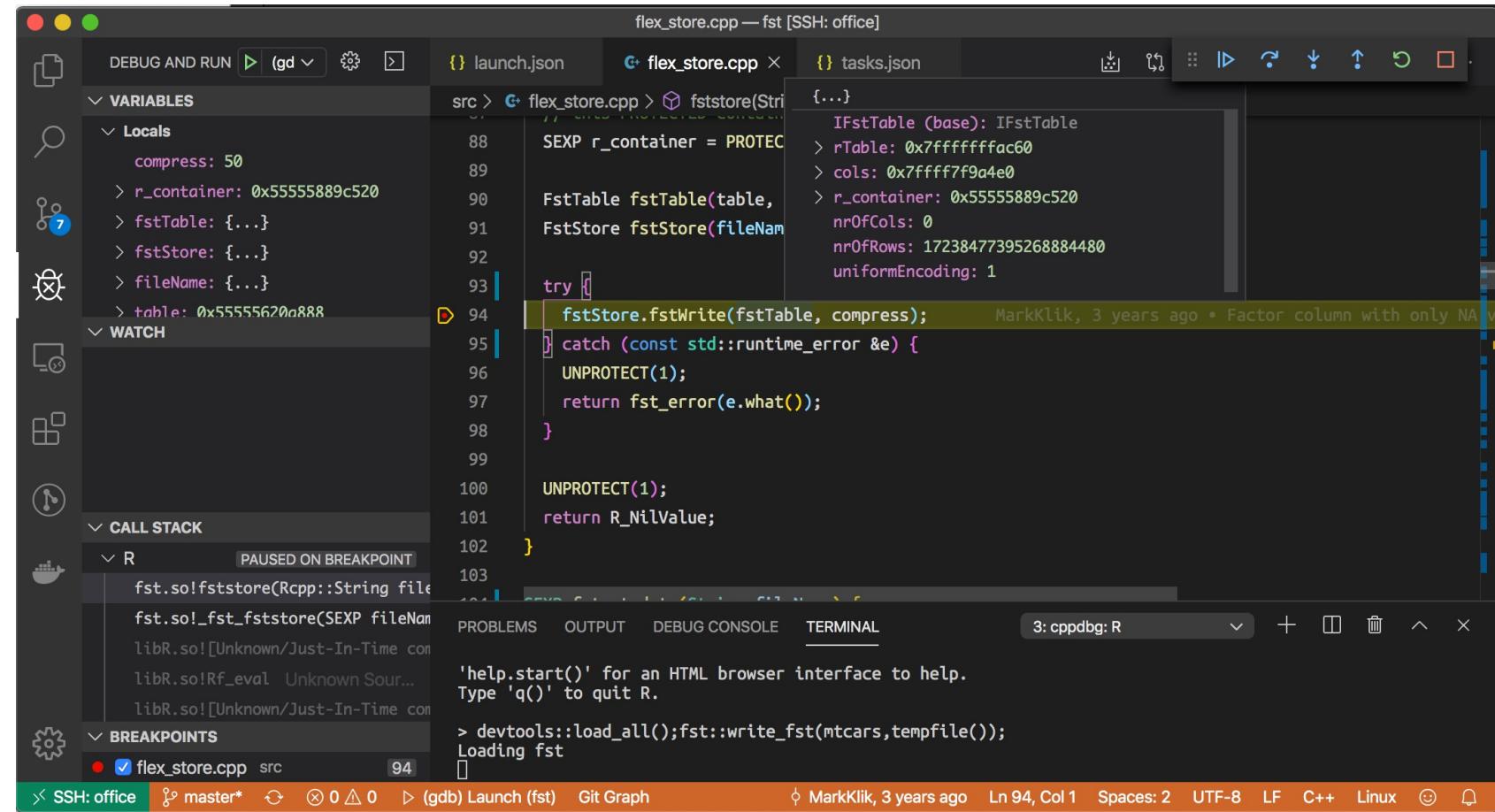
- File Explorer:** Shows the project structure under ".VS CODE-RCPP-DEMO [SSH: OFFICE]".
- Editor:** The "launch.json" file is open, showing the configuration for debugging.
- Content of launch.json:**

```
8     "name": "(gdb) Launch",
9     "type": "cppdbg",
10    "request": "launch",
11    "program": "/usr/lib/R/bin/exec/R",
12    "args": [
13        "--vanilla",
14        "-e",
15        "devtools::load_all(); devtools::test()",
16    ],
17    "stopAtEntry": false,
18    "envFile": "${workspaceFolder}/.vscode/.env",
19    "cwd": "${workspaceFolder}",
20    "externalConsole": false,
21    "MIMode": "gdb",
22    "setupCommands": [
23        {
24            "description": "Enable pretty-printing for gdb",
25            "text": "-enable-pretty-printing",
26            "ignoreFailures": true
27        }
28    ],
29    "preLaunchTask": "debug",
30    "osx": {
31        "program": "/Library/Frameworks/R.framework/Resources/bin/exec"
32    }
33 }
```

- Bottom Status Bar:** Shows the current workspace ("SSH: office"), branch ("master*"), and active editor ("(gdb) Launch (vscode-rcpp-demo)").
- Bottom Right:** A blue button labeled "Add Configuration...".

<https://github.com/renkun-ken/vscode-rcpp-demo>

Remote Development via SSH/WSL/Docker



The screenshot shows a Visual Studio Code (VS Code) interface running on a remote host via SSH. The title bar indicates the file is `flex_store.cpp` and the workspace is `fst [SSH: office]`.

The sidebar on the left contains the following sections:

- VARIABLES**: Shows a **Locals** section with variables like `compress: 50` and pointers such as `r_container: 0x55555889c520`. There is also a **WATCH** section.
- CALL STACK**: Shows a stack trace for the current thread, with the top frame being `R PAUSED ON BREAKPOINT`.
- BREAKPOINTS**: Shows a single breakpoint at line 94 of `flex_store.cpp`.

The main code editor area displays the `flex_store.cpp` file. A red dot at line 94 indicates the current position. The code uses C++11 features like `std::unique_ptr` and `std::exception`. A tooltip for the variable `r_container` is visible, showing its type as `IFstTable (base): IFstTable` and its value as `0x7fffffffac60`.

The bottom right corner of the code editor shows the status bar with the message "MarkKlik, 3 years ago • Factor column with only NA values".

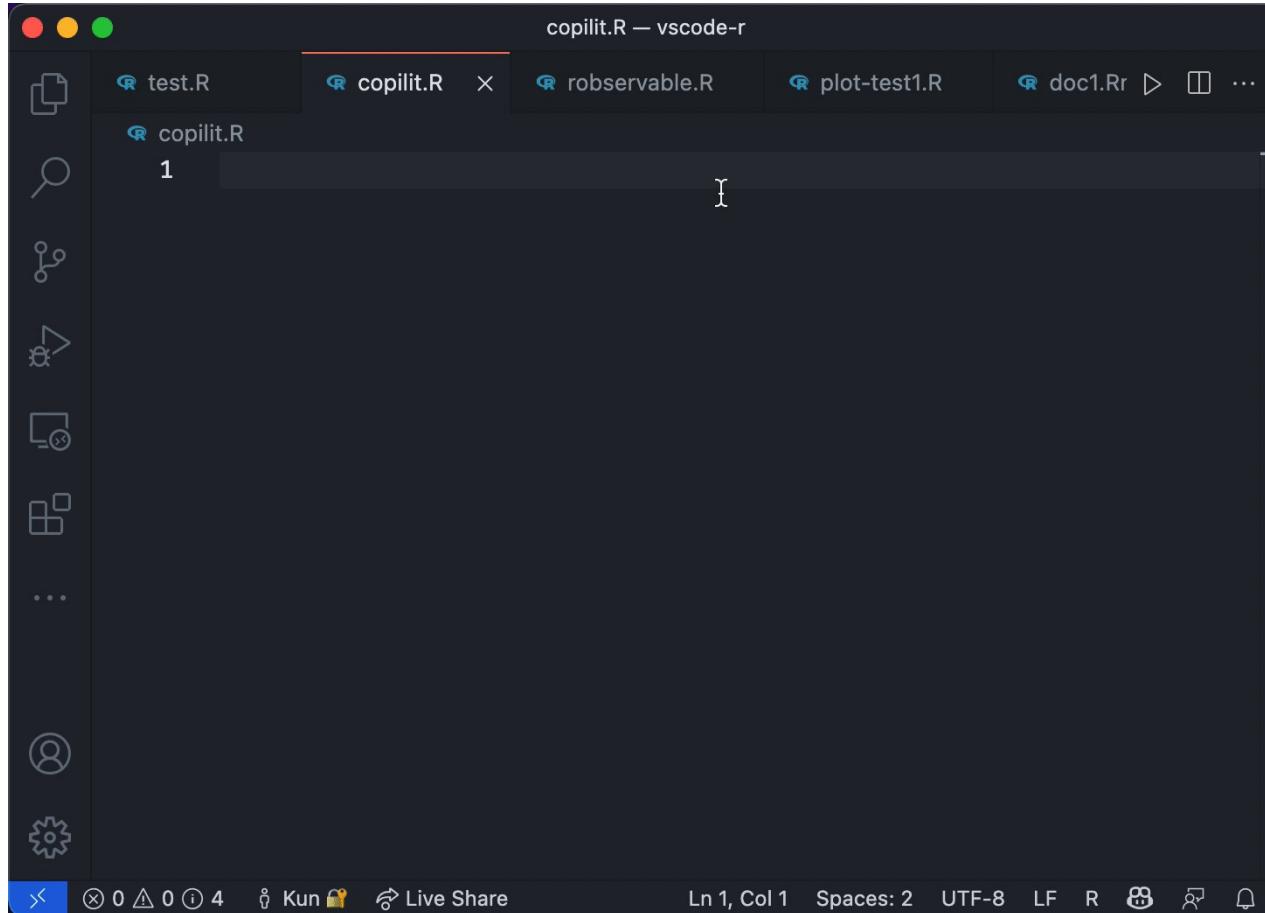
The bottom of the screen shows the VS Code navigation bar with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, displaying R command-line output:

```
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.  
  
> devtools::load_all(); fst::write_fst(mtcars, tempfile());  
Loading fst
```

The bottom status bar also shows the current file path as `3: cppdbg: R`, the line number as `Ln 94, Col 1`, and the character encoding as `UTF-8 LF`.



GitHub Copilot: Your AI pair programmer



<https://github.com/REditorSupport/languageserver>

<https://github.com/REditorSupport/vscode-R>

<https://github.com/ManuelHentschel/VSCode-R-Debugger>

<https://renkun.me>

renkun@outlook.com

github.com/renkun-ken