Troubleshooting

like a boss!

teach-shiny.rbind.io

Mine Çetinkaya-Rundel

@minebocek

mine-cetinkaya-rundel

mine@rstudio.com

write robust code

Writing robust code

Complexity is the problem; abstraction is the solution.

- Are our fragments simple enough to understand?
- Do they compose reliably?

Understandable fragments

- **Indent** your code! (Ctrl+I/Cmd+I)
- **Extract** out complicated processing logic (as opposed to UI logic) into top-level functions so you can test them separately
- Each function, reactive, observer, or module should be **small**, and do one thing
  - Function/reactive/observer bodies that don't fit on a single screen is a bad code smell
  - If you're having trouble giving something a meaningful name, maybe it's doing too much

Reliable composition

- Prefer "pure functions" — functions without side effects. Much less likely to surprise you.
  - When you do need side effects, don't put them in surprising places.
  - Consider following command-query separation — "asking a question should not change the answer"
- Reactive expressions must not have side effects
- Avoid observers and reactive values, where possible; use reactive expressions if you can help it
- For ease of reasoning, prefer: pure functional > reactive > imperative (observers)

use debugging tools

Standard R debugging tools

- Tracing
  - print()/cat()/str()
  - renderPrint eats messages, must use cat(file = stderr(), ...)
  - Also consider shinyjs package's logjs, which puts messages in the browser's JavaScript console
- Debugger
  - Set breakpoints in RStudio
  - browser()
  - Conditionals: if (!is.null(input$x)) browser()

Shiny debugging tools

- **Symptom:** Outputs or observers don't execute when expected, or execute too often
- Reactlog
  - Restart R process
  - Set options(shiny.reactlog = TRUE)
  - In the browser, Ctrl+F3 (or Cmd+F3)
- **Showcase mode:** DESCRIPTION file or runApp(display.mode = "showcase")

Shiny debugging tools

- **Symptom:** Red error messages in the UI or session abruptly terminates
- This means an R error has occurred
- Look in R console for stack traces
  - By default, Shiny hides "internal" stack traces. Use options(shiny.fullstacktrace = TRUE) if necessary to show.
- Newer versions of Shiny/Shiny Server "sanitize" errors, for security reasons (every error message is displayed as "An error has occurred")
  - See sanitizing errors article for more details, including how to view the real errors

Shiny debugging tools

- Symptom: Server logic seems OK, but unexpected/ broken/missing results in browser
- Check browser's JavaScript console for errors
- Listen in on conversation between client and server
  - options(shiny.trace=TRUE) logs messages in the R console
  - Use Chrome's Network tab to show individual websocket messages

Your turn

- Open movies_broken_01.R. It is broken in a not-very-subtle way. See if you can find and fix the bug.
- Continue on for movies_broken_02.R through movies_broken_04.R.

10m 00s

Your turn

- Open movies_broken_05.R. It is broken in a subtle way. See if you can find and fix the bug.
  - Check the box for one other type of movie and see how the text about number of movies changes.
  - Choose a low sample size and get a new sample.
  - Choose a high sample size and get a new sample.

teach common errors

"Object of type 'closure' is not subsettable"

You forgot to use () when retrieving a value from a reactive expression plot(userData) should be plot(userData())

"Unexpected symbol"
"Argument xxx is missing, with no default"

Missing or extra comma in UI. Sometimes Shiny will realize this and give you a hint, or use RStudio editor margin diagnostics.

"Operation not allowed without an active reactive context."

(You tried to do something that can only be done from inside a reactive expression or observer.)

Tried to access an input or reactive expression from directly inside the server function. You must use a reactive expression or observer instead.

Or if you really only care about the value of that input at the time that the session starts, then use isolate().

 write robust code

 use debugging tools

 teach common errors

Resources

- Debugging article on shiny.rstudio.com
- Jonathan McPherson's talk at Shiny Developer conference (video, slides)
- Hadley Wickham's Advanced R has a chapter on debugging