# Intro

## Link

<https://solutions.posit.co/guide/>

<https://posit.co/blog/migrating-from-mran-to-posit-package-manager/>

<https://github.com/r-lib/pak>

<https://posit.co/blog/challenges-in-package-management/>

https://www.accelebrate.com/library/how-to-articles/r-rstudio-library

# Shiny Applications

## Links

<https://shiny.posit.co/r/articles/build/app-formats/>

<https://shiny.posit.co/r/articles/build/two-file/>

<https://shiny.posit.co/r/articles/build/rmarkdown/>

## Examples

app02 in: Dropbox/shinny/shiny-sandbox/00\_Test\_Area/ch1\_MasteringShiny/ch1/app02

## Explanation

Shiny applications can be created using two files located in the same directory:

newDir --- ui.R

--- sever.R

The app can then be launched by calling

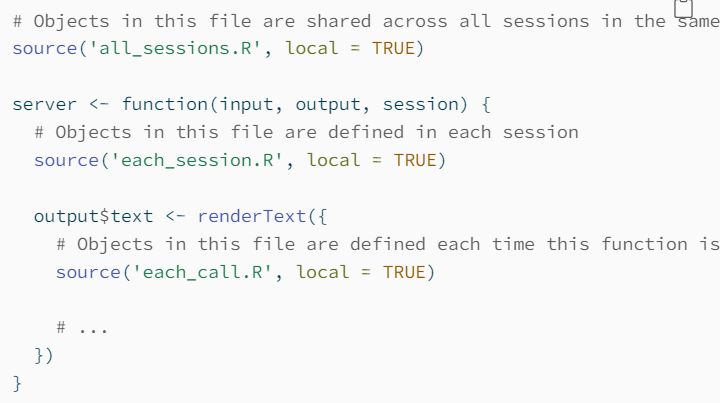
* runApp(‘newDir’) or runApp() if the call is done within the directory newDir.
* shinyAppDir(‘newDir’)
* If within the newDir a dir R is created all \*.R files are loaded without the need of a source() call in the ui and server files. Files in /R directory are loaded in alphabetical order and their functions and variables are ready to be used by ui.R or server.R
* If within the newDir there is a file called global.R it is run first of all before running ui and server funtion (<https://stackoverflow.com/questions/61861941/shinyapp-with-ui-and-server-in-separate-files>)

# Scoping

## Explanation

* The server(input, output, session) is called each time a new browser requests info from the Shiny server. Everything within this function is instantiated separately for each session.
* Data, variables and functions created in the server.R or app.R file but outside the server <- function() will be shared across sessions. A change in the variable affects all sessions but one needs to use <<- instead of <- to assign new values to it as it is outside the current session. Things work this way because app.R is sourced when one starts a Shiny app. Everything in this script is run immediately. However, the server function is only actually called when a web browser connects and a new session is started.
* Objects defined in global.R are similar to those defined in app.R outside of the server function definition, with one important difference: they are loaded into the global environment of the R session; all R code in a Shiny app is run in the global environment or a child of it.

## Example



# Versioning

## Link

<https://shiny.posit.co/r/articles/improve/upgrade-r/>

<https://stat.ethz.ch/R-manual/R-devel/library/base/html/Startup.html>

## Using renv.

https://rstudio.github.io/renv/articles/renv.html

renv::dependencies()

## Statements

* getOption("repos"). List package repositories linked to the current R implementation.
* .libPaths(). Shows the directories used as package libraries in the R implementation.
* List.files(.libPaths()). Shows the packages installed in the R implementation
* packinfo <- installed.packages(fields = c("Package", "Version")).

packinfo[,c("Package", "Version")]. Lists all installed packages

* renv::dependencies()[,c('Source','Package')]. List all packages used in the R files under the current project.
* Rig manage multiple R installations.