Learning Lab

Matt Dancho

# Overview

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| Today’s focus is on “Functionizing an Analysis” |

# Code

## Non Functionized

|  |
| --- |
|  |
| DuckDB Database + Powerpoint “template” is present in the folders  Collect will pull to memory |
|  |
|  |
|  |
|  |
|  |
|  |
| Flex table to create beautiful tables |
|  |
| Bonus Files 🡪 Creates the powerpoint |
|  |
|  |

## Tidyeval

|  |
| --- |
| Needed if your function passes arguments to other DPLYR functions.  rlang 🡪 dplyr under the hood uses rlang (uses delayed evaluation) |
| Quoting using quo() 🡪 wait till the context is right. |
| Text to symbol  Evaluation still only happens when we do eval\_tidy (typo above) |
|  |
| Function to do this programatically (here group\_by is flexible)  {{ summary\_var }} | Tells the summarize function to delay the evaluation so you don’t run into an error |
|  |
| !! if you only have 1 variable (also use “enquo”)  !!! if you possibly have more than 1 variable (also use “enquos”) |
|  |

## Functionizing the workflow

|  |
| --- |
|  |
|  |
| This is the original way (on functional). We will convert this into 3 functions |
|  |
|  |
|  |
|  |
|  |
| Reusing group variables passed earlier and using rlang””syms to convert them to symbols |
| Next learning lab will focus on converting this to a R package |