script

Turn on keyboard shortcut broadcasting

Package Structure and State

```
.libPaths()
```

back to slides

Get Ready

```
all(c("devtools", "roxygen2", "testthat", "knitr", "pkgdown") %in% installed.packages())
```

```
library(devtools)

has_devel()
```

back to slides

Create a package

```
library(devtools)
create_package("~/Desktop/libminer")
```

Explore package structure in RStudio with learners

back to slides

```
# gert::git_config()
# gert::git_config_global()

use_git_config(
  user.name = "Andy Teucher",
  user.email = "andy.teucher@gmail.com"
)
```

```
git_sitrep()
```

Git global: Name, email GitHub user: PAT discovered, User, email(s)

```
use_git()
```

back to slides

```
usethis::use_devtools()
```

Restart R - check that devtools is loaded

back to slides

```
use_r("lib-summary")
```

```
lib_summary <- function() {
  pkgs <- utils::installed.packages()
  pkg_tbl <- table(pkgs[, "LibPath"])
  pkg_df <- as.data.frame(pkg_tbl, stringsAsFactors = FALSE)
  names(pkg_df) <- c("Library", "n_packages")
  pkg_df
}
```

back to slides

```
load_all()

lib_summary()
```

Commit

back to slides

```
check()
```

back to slides

Open DESCRIPTION file

```
use_mit_license()
```

Commit

back to slides

DESCRIPTION file:

```
Package: libminer
Title: Explore Your R Libraries
Version: 0.0.0.9000
Authors@R:
    person("Andy", "Teucher", , "andy.teucher@gmail.com", role = c("aut", "cre"),
           comment = c(ORCID = "0000-0002-7840-692X"))
Description: Provides functions for learning about your R libraries, and the
    packages you have installed.
```

back to slides

```
check()
```

Commit

back to slides

Tell your git repository where its corresponding repository on GitLab is

```
git remote add origin https://gitlab.com/boshek/foo.git
```

Push all your committed code to that repository

At this step you may be required to enter in your GitLab username and password.

```
git push --set-upstream origin main
```

Go to GitLab page, explore show clone of local

back to slides

```
edit_r_profile()
```

put this in the chat

```
# Set usethis options
#
options(
  usethis.description = list(
    "Authors@R" = utils::person(
      "Jane", "Doe",
      email = "jane@example.com",
      role = c("aut", "cre"),
      comment = c(ORCID = "0000-1111-2222-3333")
    )
  )
)

options(
  usethis.description = list(
    "Authors@R" = utils::person(
      "Andy", "Teucher",
      email = "andy.teucher@gmail.com",
      role = c("aut", "cre"),
      comment = c(ORCID = "0000-0002-7840-692X")
    )
  )
)

options(
  warnPartialMatchArgs = TRUE,
  warnPartialMatchDollar = TRUE,
  warnPartialMatchAttr = TRUE
)
```

back to slides

Documentation

`Ctrl + .`

`Ctrl+Alt+Shift+R`

```
#' R Library Summary
#'
#' Provides a brief summary of the package libraries on your machine
#'
#' @return A data.frame containing the count of packages in each of the user's
#'   libraries
#' @export
#'
#' @examples
#' lib_summary()
```

```
document()
```

Go to `man/lib_summary.Rd`

```
load_all()

?lib_summary
```

```
check()
```

Look at NAMESPACE

commit

back to slides

Package-level documentation

```
use_package_doc()

document()
```

Go to `man/libminer2-package.Rd`

Preview and check again

```
load_all()

?libminer

check()
```

back to slides

```
install()
library(libminer)

lib_summary() # note one more package than before - that's yours!
```

commit and push

Testing

restart R

```
use_testthat()
```

Have `R/lib-summary.R` open

```
use_test()
```

```
test_that("lib_summary returns expected results", {
  res <- lib_summary()
  expect_s3_class(res, "data.frame")
  expect_equal(ncol(res), 2)
  expect_equal(names(res), c("Library", "n_packages"))
  expect_type(res$Library, "character")
  expect_type(res$n_packages, "integer")
})

test_that("lib_summary fails appropriately", {
  expect_error(lib_summary("foo"), "unused argument")
})
```

```
test()

check()
```

reinforce file structure

commit

back to slides

Dependencies

```
use_package("fs")
```

Look at DESCRIPTION, NAMESPACE (no change)

commit

```
lib_summary <- function(sizes = FALSE) {
  if (!is.logical(sizes)) {
    stop("'sizes' must be logical (TRUE/FALSE).")
  }

  pkgs <- utils::installed.packages()
  pkg_tbl <- table(pkgs[, "LibPath"])
  pkg_df <- as.data.frame(pkg_tbl, stringsAsFactors = FALSE)
  names(pkg_df) <- c("Library", "n_packages")

  if (sizes) {
    pkg_df$lib_size <- vapply(
      pkg_df$Library,
      function(x) {
        sum(fs::file_size(fs::dir_ls(x, recurse = TRUE)))
      },
      FUN.VALUE = numeric(1)
    )
  }
  pkg_df
}
```

```
test() # failure for unused argument
```

```
test_that("lib_summary fails appropriately", {
  expect_error(lib_summary(sizes = "foo"), "'sizes' must be logical")
})

test_that("sizes argument works", {
  res <- lib_summary(sizes = TRUE)
  expect_equal(names(res), c("Library", "n_packages", "lib_size"))
  expect_type(res$lib_size, "double")
})
```

commit

```
check() # will warn about undocumented parameter
```

Ctrl+Alt+Shift+R will insert the spot for the sizes param

```
#' Provides a brief summary of the package libraries on your machine
#'
#' @param sizes Should the sizes of the libraries be calculated?
#'   Logical; default `FALSE`.
#'
#' @return A data.frame containing the count of packages in each of the user's
#'   libraries. A `lib_size` column is included if `sizes = TRUE`.
#' @export
#'
#' @examples
#' lib_summary()
#' lib_summary(sizes = TRUE)
```

```
document()

check()
```

Test it out

```
load_all()

?lib_summary

lib_summary(sizes = TRUE)
```

commit

back to slides

```
use_import_from("purrr", "map_dbl")
```

Look at: DESCRIPTION, R/libminer-package.R, NAMESPACE

```
if (sizes) {
  pkg_df$lib_size <- map_dbl(
    pkg_df$Library,
    ~ sum(fs::file_size(fs::dir_ls(.x, recurse = TRUE))),
  )
}
```

```
test()
```

Note importance of tests here

```
check()
```

commit and push

README

```
use_readme_rmd()
```

```
---
output: github_document
---

<!-- README.md is generated from README.Rmd. Please edit that file -->



# libminer

<!-- badges: start -->

<!-- badges: end -->
```

````
The goal of libminer is to provide an overview of your R library setup. It is a toy
package created as a part of a workshop and not meant for serious use.

## Installation

You can install the development version of libminer from [GitLab](https://gitlab.com/) with:

```r
# install.packages("devtools")
devtools::install_gitlab("ateucher/libminer")
```

## Example usage

To get a count of installed packages in each of your library locations,
optionally with the total sizes, use the `lib_summary()` function:

```{r example}
library(libminer)
lib_summary()
# specify `sizes = TRUE` to calculate the total size on disk of your packages
lib_summary(sizes = TRUE)
```
````

```
build_readme()
```

```
check()

install()
```

commit

return to slides

Continuous Integration

```
use_gitlab_ci()
```

commit and push

return to slides

Vignettes

```
use_vignette("lib-sitrep", "Package Library Situation Report")
```

Releasing

```
use_news_md()
```

Bonus - factoring out an internal function

First refactor

- designed to fail CI so don't check() before commiting and pushing

```
#' R Library Summary
#'
#' Provides a brief summary of the package
#' libraries on your machine
#'
#' @param sizes a logical indicating whether or not to calculate
#'    sizes
#'
#' @return A data.frame containing the count
#'    of packages in each of your libraries.
#' @export
#'
#' @examples
#' lib_summary()
#' lib_summary(sizes = TRUE)
lib_summary <- function(sizes = FALSE) {
  pkgs <- utils::installed.packages()
  pkg_tbl <- table(pkgs[, "LibPath"])
  pkg_df <- as.data.frame(pkg_tbl, stringsAsFactors = FALSE)
  names(pkg_df) <- c("library", "n_packages")

  if (sizes) {
    pkg_df <- calculate_sizes(pkg_df)
  }
```

```
  pkg_df
}

#' calculate sizes
#'
#' @param df a data.frame
#'
#' @return df with a lib_size column
#' @noRd
calculate_sizes <- function(df) {
  df$lib_size <- map_dbl(
    df$library,
    ~ sum(file_size(dir_ls(.x, recurse = TRUE)))
  )
  df
}
```

```
#' calculate sizes
#'
#' @param df a data.frame
#'
#' @return df with a lib_size column
#' @noRd
calculate_sizes <- function(df) {
  df$lib_size <- purrr::map_dbl(
    df$library,
    ~ sum(fs::file_size(fs::dir_ls(.x, recurse = TRUE)))
  )
  df
}
```