Add data and files to R packages

What else can I do with an R package?

What else can I do with an R package?

Store data and files

What else can I do with an R package?

Store data and files

Create R Markdown and other templates

What else can I do with an R package?

Store data and files

Create R Markdown and other templates

Launch and store Shiny apps

Data in packages

data in ggplot2

```
## # A tibble: 11 × 2
##    data           description
##    <chr>          <chr>
##  1 diamonds       Prices of over 50,000 round cut diamonds
##  2 economics      US economic time series
##  3 economics_long US economic time series
##  4 faithfuld      2d density estimate of Old Faithful data
##  5 luv_colours    'colors()' in Luv space
##  6 midwest        Midwest demographics
##  7 mpg            Fuel economy data from 1999 to 2008 for 3…
##  8 msleep         An updated and expanded version of the ma…
##  9 presidential   Terms of 11 presidents from Eisenhower to…
## 10 seals          Vector field of seal movements
## 11 txhousing      Housing sales in TX
```

Data packages

data in babynames

```
## # A tibble: 4 × 2
##   data       description
##   <chr>      <chr>
## 1 applicants Applicants.
## 2 babynames  Baby names.
## 3 births     Births
## 4 lifetables Lifetables
```

Data packages

data in [claremontrun]


```
## # A tibble: 7 × 2
##   data                  description
##   <chr>                 <chr>
## 1 character_visualization Counts of character speech, thoug…
## 2 characters            Descriptions of character actions…
## 3 comic_bechdel         Whether or not an issue of anothe…
## 4 covers                Data on covers of issues of Uncan…
## 5 issue_collaborators   Data about other collaborators on…
## 6 locations             Locations of issues in the Clarem…
## 7 xmen_bechdel          Whether or not an issue of Uncann…
```

Add data to your package

1. Include raw data and cleaning scripts using `use_data_raw()`
2. Clean the data in the file created above, then use `use_data()`
3. For non-exported data, `use_data(internal = TRUE)`
4. Document your data

Data dictionary

```
data_dictionary <-
  tibble::tibble(
    database = "daily_active_users",
    variable = c("users", "sector"),
    description = c(
      "Active reactor-using households",
      "Midgar Sector #"
    ),
  )
```

```
use_data_raw("data_dictionary")
```

```
shinRa
├── .Rbuildignore
├── .gitignore
├── DESCRIPTION
├── NAMESPACE
├── R/
│   ├── themes.R
├── data-raw
│   └── data_dictionary.R
├── man
│   ├── theme_mako.Rd
├── tests
│   ├── testthat
│   │   └── test-themes.R
│   ├── testthat.R
├── vignettes
│   └── intro-to-shinRa.Rmd
└── shinRa.Rproj
```

use_data_raw("data_dictionary")

data_dictionary.R

```
## code to prepare `data_dictionary` dataset goes here

usethis::use_data(data_dictionary, overwrite = TRUE)
```

use_data_raw("data_dictionary")

data_dictionary.R

```
## code to prepare `data_dictionary` dataset goes here

data_dictionary <-
  tibble::tibble(
    database = "daily_active_users",
    variable = c("users", "sector"),
    description = c(
      "Active reactor-using households",
      "Midgar Sector #"
    ),
  )

usethis::use_data(data_dictionary, overwrite = TRUE)
```

use_data_raw("data_dictionary")

data_dictionary.R

```r
## code to prepare `data_dictionary` dataset goes here

data_dictionary <-
  tibble::tibble(
    database = "daily_active_users",
    variable = c("users", "sector"),
    description = c(
      "Active reactor-using households",
      "Midgar Sector #"
    ),
  )

usethis::use_data(data_dictionary, overwrite = TRUE)
```

use_data(data_dictionary)

```
shinRa
├── .Rbuildignore
├── .gitignore
├── DESCRIPTION
├── NAMESPACE
├── R/
│   ├── themes.R
├── data
│   └── data_dictionary.rda
├── data-raw
│   └── data_dictionary.R
├── man
│   ├── theme_mako.Rd
├── tests
│   ├── testthat
│   │   └── test-themes.R
│   └── testthat.R
├── vignettes
│   └── intro-to-shinRa.Rmd
└── shinRa.Rproj
```

```
library(shinRa)
data_dictionary
```

```
## # A tibble: 2 × 3
##   database           variable description
##   <chr>              <chr>    <chr>
## 1 daily_active_users users    Active reactor-using househol…
## 2 daily_active_users sector   Midgar Sector #
```

Your Turn 1

Let's create a data dictionary for avalanche and add the results to a vignette. First, run `use_data_raw()`. Call the data "data_dictionary". In the data processing file, put this code before the `use_data()` line. Then, source the script. This will run `use_data()` for you.

Open `R/tables.R`. Add a function called `gt_data_dictionary()` that wraps `data_dictionary` in `gt::gt()`. You can use `gt_donations()` as a starting point for the new function.

Re-document and re-build the package.

Open `vignettes/data-dictionary.Rmd`. On line 24, run `gt_data_dictionary()` and knit the vignette.

Your Turn 1

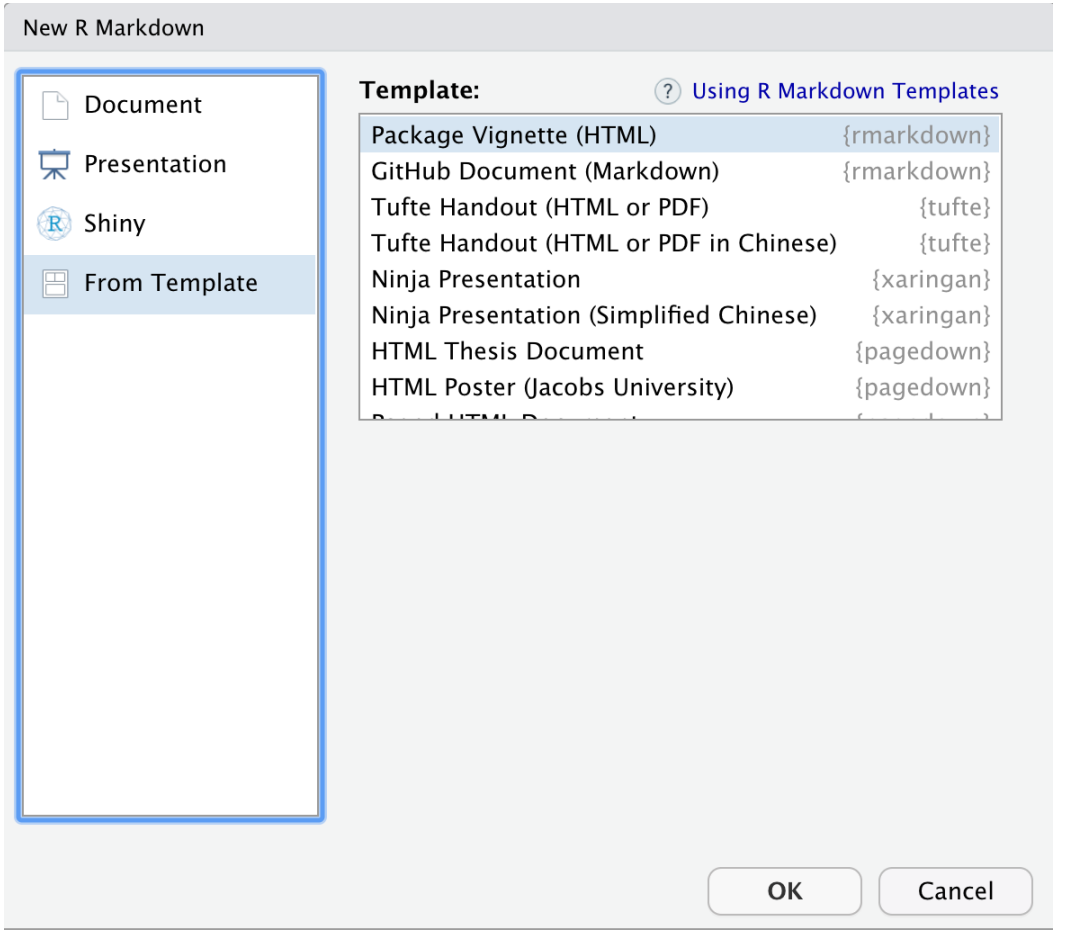
```
use_data_raw("data_dictionary")
source("data-raw/data_dictionary.R")
```

```
#' Create gt table for the data dictionary
#'
#' @return a `gt` table
#' @export
gt_data_dictionary <- function() {
  gt::gt(data_dictionary)
}
```

Including data for tests

1. Hard-code it in a `helper-` or `test-` file.
2. Put a (small) file in `tests/testthat`
3. Use internal data (`use_data(internal = TRUE)`)
4. Use exported data (`use_data()`)

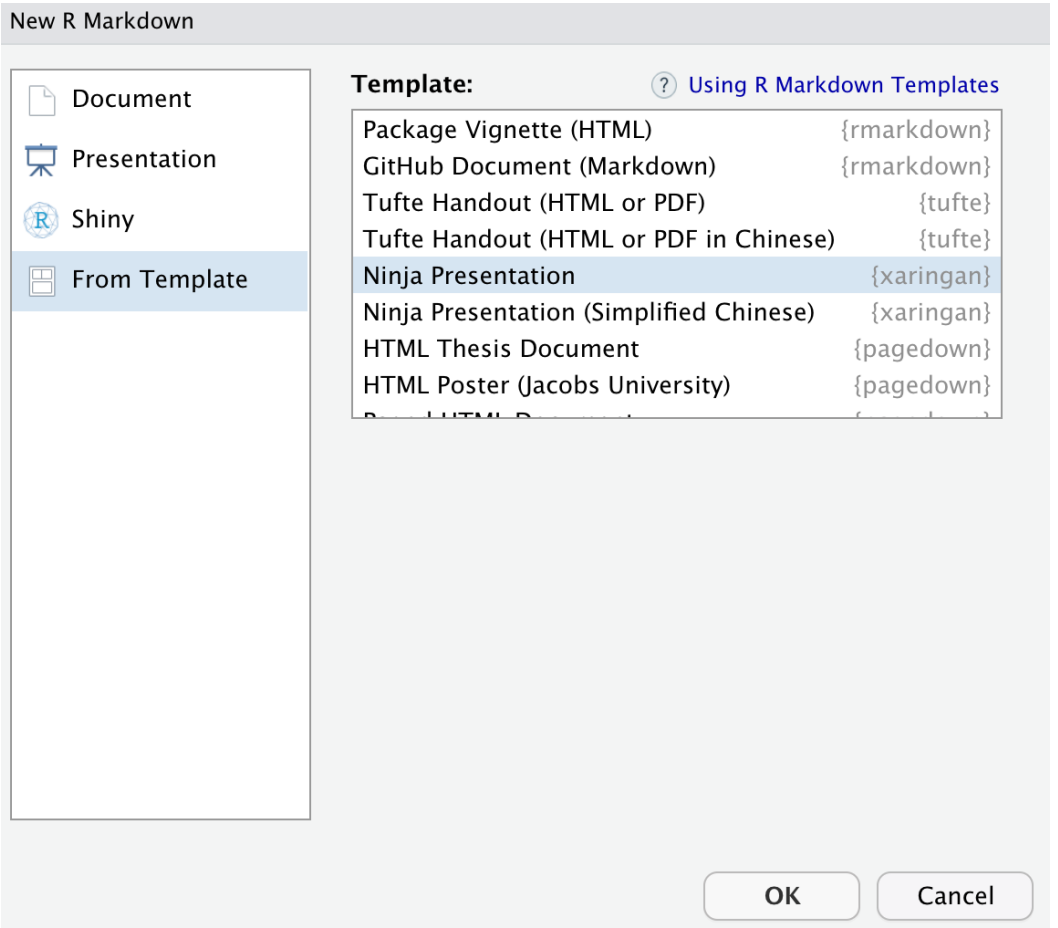
R Markdown Templates



```
use_rmarkdown_template("report")
```

```
shinRa
├── .Rbuildignore
├── .gitignore
├── DESCRIPTION
├── NAMESPACE
├── R/
│   ├── themes.R
├── data-raw
│   └── data_dictionary.R
├── inst
│   └── rmarkdown
│       └── templates
│           └── report
│               ├── skeleton
│               │   └── skeleton.Rmd
│               └── template.yaml
├── man
│   ├── theme_mako.Rd
├── tests
│   ├── testthat
│   │   └── test-themes.R
│   └── testthat.R
├── vignettes
│   └── intro-to-shinRa.Rmd
└── shinRa.Rproj
```

xaringan template



xaringan template: `template.yaml`

```
name: Ninja Presentation
description: >
  Create a presentation using the xaringan package based
  on remark.js.
create_dir: false
```

xaringan template: `skeleton.Rmd`

```
title: "Presentation Ninja"
subtitle: "⚔ <br/>with xaringan"
author: "Yihui Xie"
institute: "RStudio, Inc."
date: "2016/12/12 (updated: `r Sys.Date()`)"
output:
  xaringan::moon_reader:
    lib_dir: libs
    nature:
      highlightStyle: github
      highlightLines: true
      countIncrementalSlides: false
---

background-image: url(https://upload.wikimedia.org/wikipedia/commons/b/be/Sharingan_triple.svg)

```{r setup, include=FALSE}
options(htmltools.dir.version = FALSE)
```

???

Image credit: [Wikimedia Commons](https://commons.wikimedia.org/wiki/File:Sharingan_triple.svg)

---
class: center, middle

xaringan

/ʃaː.'riŋ.gan/

---
class: inverse, center, middle

Get Started
```

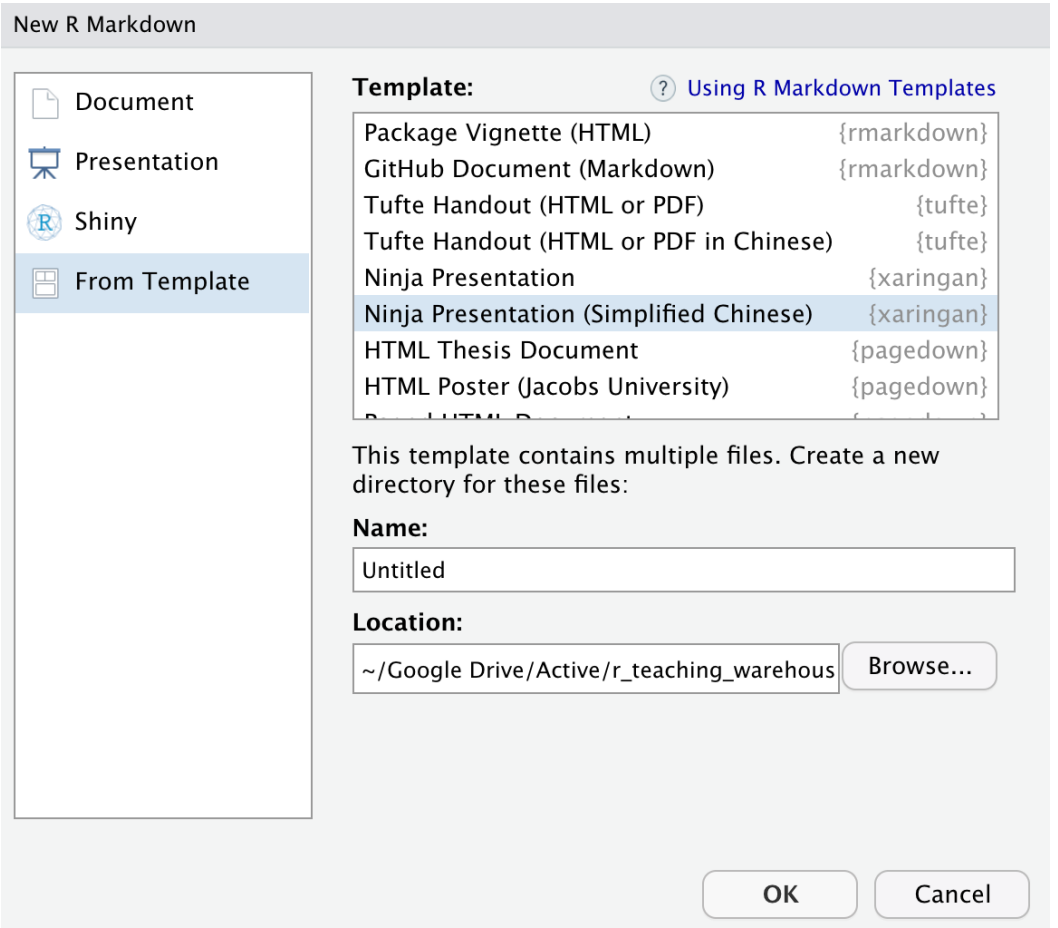
xaringan Chinese template: `template.yaml`

```
name: Ninja Presentation (Simplified Chinese)
description: >
  Create a presentation using the xaringan package based
  on remark.js.
  create_dir: true
```

xaringan Chinese template

```
## inst/rmarkdown/templates/xaringan_zh-CN/skeleton/
## ├── skeleton.Rmd
## └── zh-CN.css
```

xaringan Chinese template



Your Turn 2

Create a new R Markdown template called "avalanche-report" using `use_rmarkdown_template()`. For the `template_description` argument, set it to: "A weekly report of AVALANCHE activities."

Inside the `exercises/` folder, there is a file called `report.Rmd`. Copy and paste its contents into `inst/rmarkdown/templates/avalanche-report/skeleton/skeleton.Rmd`, which you just created.

Re-build the package (`build()` or `Cmd/Ctrl + Shift + B`).

Create a new R Markdown document using `File > New File > R Markdown...`. Then, find your template under the "From Template" tab. Open it.

Your Turn 2

```
use_rmarkdown_template(
  "template_description",
  "A weekly report of AVALANCHE activities."
)
```

On `inst/`

We can put *any* arbitrary file in `inst/`.

On `inst/`

We can put any arbitrary file in `inst/`.

Be careful not to name the folders the same as any of the top-level package folders, e.g. `R/` or `data/`.

On `inst/`

We can put any arbitrary file in `inst/`.

Be careful not to name the folders the same as any of the top-level package folders, e.g. `R/` or `data/`.

Access content with *`system.file()`* or *`fs::path_package()`*

system.file()

```
system.file(
  file.path("rmarkdown", "templates", "report.Rmd"),
  package = "shinRa",
  mustWork = TRUE
)
```

system.file()

```
system.file(
  file.path("rmarkdown", "templates", "report.Rmd"),
  package = "shinRa",
  mustWork = TRUE
)
```

```
## [1] "path/to/shinRa/rmarkdown/templates/report.Rmd"
```

Other templates

Other templates

Analysis workflows or projects

Other templates

Analysis workflows or projects

More detailed R Markdown templates

Other templates

Analysis workflows or projects

More detailed R Markdown templates

YAML and other configuration files

Templates with usethis

```
use_cran_comments()
```

Templates with usethis

`use_cran_comments()`

```
use_cran_comments <- function(open = interactive()) {
  check_is_package("use_cran_comments()")
    use_template(
      "cran-comments.md",
      data = list(rversion = glue("{version$major}.{version$minor}")),
      ignore = TRUE,
      open = open
    )
}
```

Templates with usethis

`use_cran_comments()`

```
use_cran_comments <- function(open = interactive()) {
  check_is_package("use_cran_comments()")
    use_template(
      "cran-comments.md",
      data = list(rversion = glue("{version$major}.{version$minor}")),
      ignore = TRUE,
      open = open
    )
}
```

use_template() and inst/templates

```
## # A tibble: 55 × 2
##    path                type
##    <chr>               <fct>
##  1 CODE_OF_CONDUCT.md  file
##  2 Jenkinsfile         file
##  3 Makefile            file
##  4 NEWS.md             file
##  5 addins.dcf          file
##  6 appveyor.yml        file
##  7 article.Rmd         file
##  8 circleci-config.yml file
##  9 citation-template.R file
## 10 code-cpp11.cpp      file
## # … with 45 more rows
```

cran-comments.md

```
# ## Test environments
# * local R installation, R {{{ rversion }}}
# * ubuntu 16.04 (on travis-ci), R {{{ rversion }}}
# * win-builder (devel)
#
# ## R CMD check results
#
# 0 errors | 0 warnings | 1 note
#
# * This is a new release.
```

whisker

```
whisker::whisker.render(
"## Test environments
  * local R installation, R {{{ rversion }}}
  * ubuntu 16.04 (on travis-ci), R {{{ rversion }}}
 * win-builder (devel)

 ## R CMD check results

 0 errors | 0 warnings | 1 note

 * This is a new release.",
data = list(rversion = glue::glue("{version$major}.{version$minor}"))
)
```

whisker

```
## ## Test environments
##  * local R installation, R 4.1.1
##  * ubuntu 16.04 (on travis-ci), R 4.1.1
##  * win-builder (devel)
##
##  ## R CMD check results
##
##  0 errors | 0 warnings | 1 note
##
##  * This is a new release.
```

Creating a template: `inst/templates/config.yml`

```
edit_template("config.yml")

# run without arguments to open a menu of existing templates
edit_template()
```

In `config.yml`:

```
name: {{{name}}}
  affiliation: Shinra, Inc.
description: {{{description}}}
```

```
use_data_raw("data_dictionary")
```

```
shinRa
├── .Rbuildignore
├── .gitignore
├── DESCRIPTION
├── NAMESPACE
├── R/
│   ├── themes.R
├── data-raw
│   └── data_dictionary.R
├── man
│   ├── theme_mako.Rd
├── tests
│   ├── testthat
│   │   └── test-themes.R
│   ├── testthat.R
├── vignettes
│   └── intro-to-shinRa.Rmd
└── shinRa.Rproj
```

Wrapping `use_template()`

```
use_yaml_template <- function(path, name, description) {
  usethis::use_template(
    "config.yml",
    save_as = file.path(path, "config.yml"),
    package = "shinRa",
    data = list(name = name, description = description)
  )
}
```

Create a template file

```
name: {{{name}}}
  affiliation: Shinra, Inc.
description {{{description}}}
```

Create a wrapper function

```
use_yaml_template <- function(path, name, description) {
  usethis::use_template(
    "config.yml",
    save_as = file.path(path, "config.yml"),
    package = "shinRa",
    data = list(name = name, description = description)
  )
}
```

```
name: {{{name}}}
  affiliation: Shinra, Inc.
description {{{description}}}
```

whisk data

```
use_yaml_template <- function(path, name, description) {
  usethis::use_template(
    "config.yml",
    save_as = file.path(path, "config.yml"),
    package = "shinRa",
    data = list(name = name, description = description)
  )
}
```

```
use_yaml_template(
  ".",
  "Malcolm Barrett",
  "A project to analyze reactor data"
)
```

```
name: Malcolm Barrett
  affiliation: Shinra, Inc.
description: A project to analyze reactor data
```

Your Turn 3

Let's create a usethis-style function to help set up analysis projects. In the folder `exercises/templates/`, there are three files: "packages.R", "analysis.R", and "report.Rmd". Open them up and take a look. Note that `report.Rmd` has some whisker variables in the YAML!

Copy the files to `inst/templates` by running this code in the console:
`fs::dir_copy("exercises/templates/", "inst/templates")`

Open `R/create_analysis.R`. `create_analysis()` is going to help us set up the project directory, but we need to complete it. In lines 22, 23, and 25, add the template names: "packages.R", "analysis.R", and "report.Rmd".

Remember that "report.Rmd" has data to whisk. We need to tell it what to pass to the final file. The `data` argument takes a named list. For this argument, write: `list(author = author, title = title)`.

Below `create_analysis()` is a helper function, `usethis::use_template()`, to create files from templates. Change the `package` argument to "avalanchr".

Your Turn 3

```
create_analysis <- function(path = ".", folder = "avalanche_analysis'
  analysis_path <- fs::path(path, folder)
  if (fs::dir_exists(analysis_path)) fs::dir_delete(analysis_path)

  usethis::ui_done("Writing {usethis::ui_path(folder)}")
  fs::dir_create(analysis_path)

  use_avalanche_template("packages.R", folder = folder)
  use_avalanche_template("analysis.R", folder = folder)
  use_avalanche_template(
    "report.Rmd",
    folder = folder,
    data = list(author = author, title = title)
  )

  invisible(analysis_path)
}
```

Your Turn 3

```
use_avalanche_template <- function(template, folder, data = list()) {
  usethis::use_template(
    template = template,
    save_as = fs::path(folder, template),
    data = data,
    package = "avalanchr"
  )
}
```

```
create_analysis()
```

Shiny apps

Add shiny app to `inst/shinyapps/` (or something similar)

Shiny apps

Add shiny app to `inst/shinyapps/` (or something similar)

Then, use an R function like this to launch it:

```
launch_app <- function(app) {
  app_dir <- system.file(
    "shinyapps",
    app,
    package = "shinRa",
    mustWork = TRUE
  )
  shiny::runApp(app_dir)
}
```

Shiny apps

Add shiny app to `inst/shinyapps/` (or something similar)

Then, use an R function like this to launch it:

```
launch_app <- function(app) {
  app_dir <- system.file(
    "shinyapps",
    app,
    package = "shinRa",
    mustWork = TRUE
  )
  shiny::runApp(app_dir)
}
```

Shiny apps

Add shiny app to `inst/shinyapps/` (or something similar)

Then, use an R function like this to launch it:

```
launch_app <- function(app) {
  app_dir <- system.file(
    "shinyapps",
    app,
    package = "shinRa",
    mustWork = TRUE
  )
  shiny::runApp(app_dir)
}
```

See "Supplementing your R package with a Shiny app" for more.

Your Turn 4

There is an app in the `examples/` folder called `plot_app/`. It has two Shiny files: `ui.R` and `server.R`.

Run `use_package("shiny")`

Create a folder in `inst/` called `shinyapps`

Create an R file with `use_r()` called "launch_app".

Write a function to launch the app

Add a roxygen skeleton and add a title and describe any parameters you used (if you used any)

Re-build your package and launch the app with your function

Your Turn 4

```
#' Launch Reactor Data Shiny App
#'
#' @return a shiny app
#' @export
launch_app <- function() {
  app_dir <- system.file(
    "shinyapps",
    "shiny_reactor_report",
    package = "avalanchr",
    mustWork = TRUE
  )

  shiny::runApp(app_dir)
}
```

```
launch_app()
```

With `inst/`, the world is yours

With `inst/`, the world is yours

Source files with `inst/scripts` and `sys.source()`

With `inst/`, the world is yours

Source files with `inst/scripts` and `sys.source()`

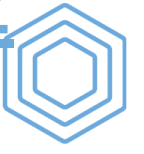
Add citations to `inst/CITATION` (`use_citation()`)

With `inst/`, the world is yours

Source files with `inst/scripts` and `sys.source()`

Add citations to `inst/CITATION` (`use_citation()`)

Much more. See R Packages, ed 2



Ignoring files at the top-level

`use_build_ignore()`: don't include when building R package

Ignoring files at the top-level

`use_build_ignore()`: don't include when building R package

`use_git_ignore()`: don't commit (credentials, certain rendered files, etc)

Ignoring files at the top-level

`use_build_ignore()`: don't include when building R package

`use_git_ignore()`: don't commit (credentials, certain rendered files, etc)

Try `git_vaccinate()`

RStudio Add-ins

Small shiny apps using `miniui` and RStudio.

Create one with `use_addin()`

See "Shiny Gadgets" and "Designing Gadget UI"