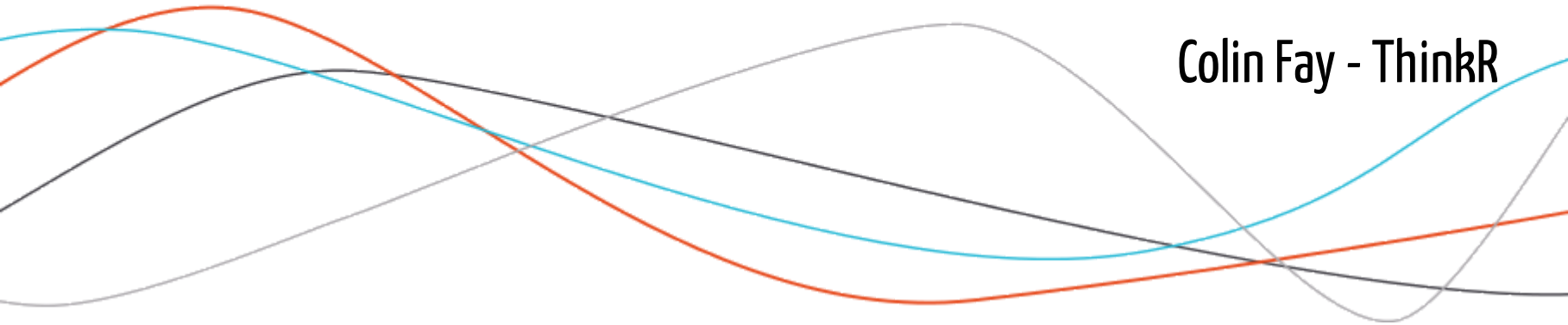
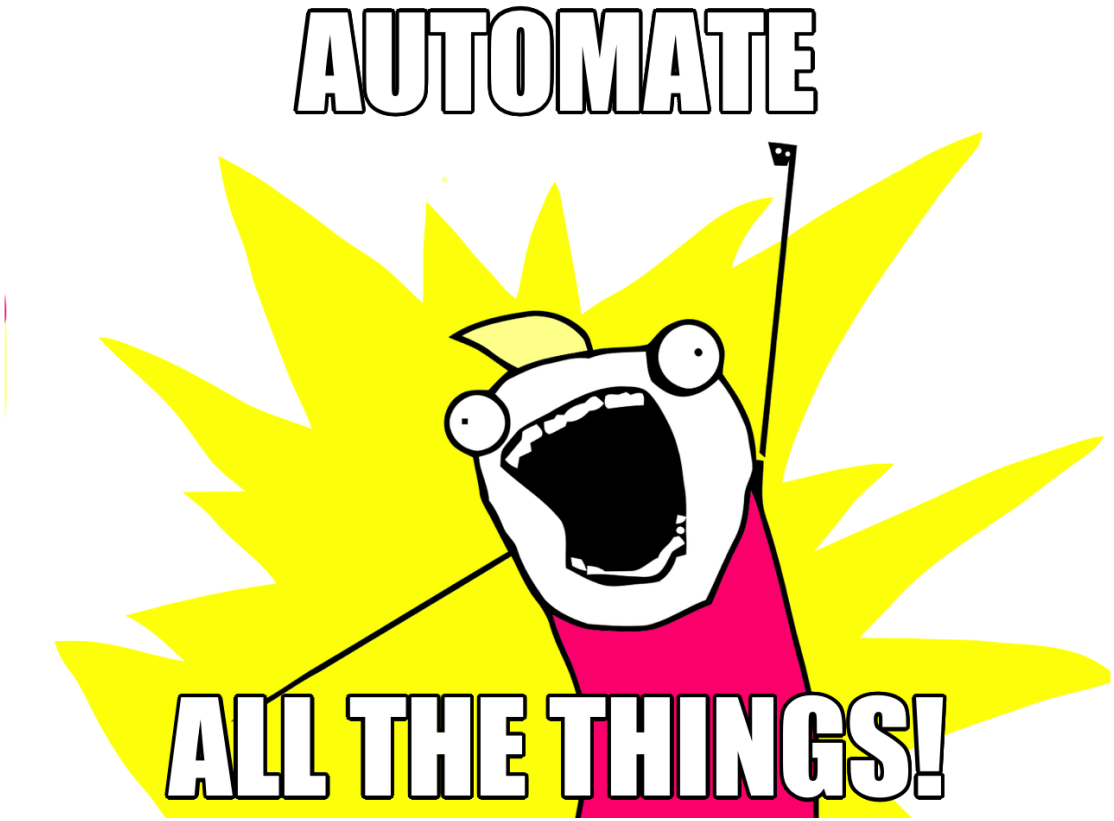
Hacking RStudio

Part 3: templates



Colin Fay - ThinkR

Templates



Templates - Why ?

- Automation of common markdown formats
- Sharing templates

> For example, ThinkR has an internal package to produce slides (theses slides are generated through a customized `{xaringan}` format)

Examples

- `{pagedown}` - Paginate the HTML Output of R Markdown with CSS for Print
- `{xaringan}` - Presentation Ninja 幻灯忍者 · 写轮眼
- `{rticles}` - LaTeX Journal Article Templates for R Markdown
- `{tufte}` - Tufte Styles for R Markdown Documents

Markdown template

- Create a package
- `usethis::use_rmarkdown_template("mymarkdown")`
- update `inst/rmarkdown/templates/mymarkdown/template.yaml` & `inst/rmarkdown/templates/mymarkdown/skeleton/skeleton.Rmd`
- Install the package
- Find your template File > New File > RMarkdown > From Template

template.yaml

```
name: mymarkdown
description: >
   A description of the template
create_dir: FALSE
```

skeleton.Rmd

````
---
title: "Template Title"
author: "Your Name"
date: "The Date"
output: output_format
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Adding an RMarkdown Template

This file is what a user will see when they select your template. Make sure
that you update the fields in the yaml header. In particular you will want
to update the `output` field to whatever format your template requires.

This is a good place to demonstrate special features that your template
provides. Ideally it should knit out-of-the-box, or at least contain clear
instructions as to what needs changing.

Finally, be sure to remove this message!
````

Include external files along the Rmd

- Add files to the `skeleton` folder
- Turn `create_dir` to `TRUE` in the `yaml`

Project Templates

- Create a package
- Create a function to launch on project creation
- Define template metadata
- Input Widgets

Project Templates

```
create_golem <- function(path, ...) {

  dir.create(path, recursive = TRUE, showWarnings = FALSE)

  ll <- list.files(
    path = golem_sys("shinyexample"),
    full.names = TRUE, all.files = TRUE, no.. = TRUE
  )

  file.copy(
    from = ll,
    to = path,
    overwrite = TRUE,
    recursive = TRUE
  )

}
```

Project Templates

`inst/rstudio/templates/project/create_golem.dcf`

```
Binding: create_golem_gui
Title: Package for Shiny App using golem
OpenFiles: dev/01_start.R
Icon: golem.png
```

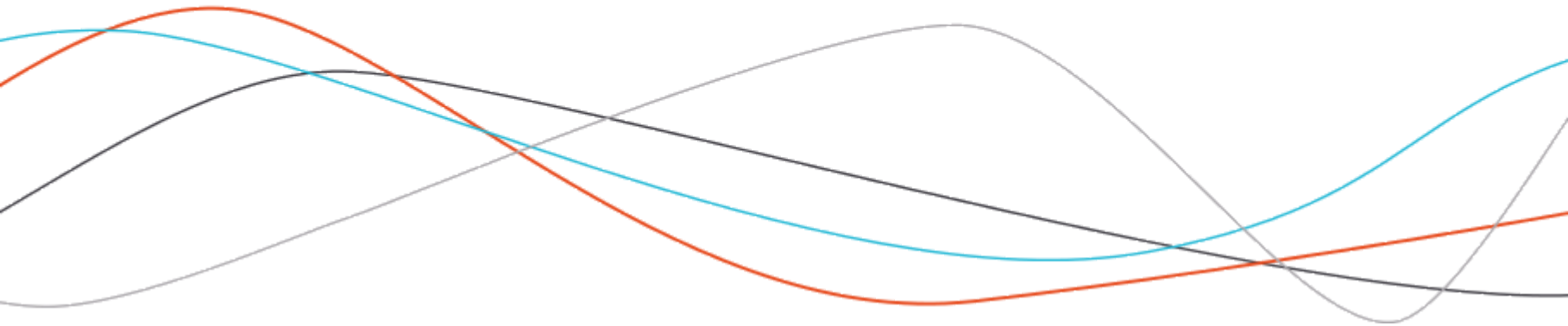
- Binding: the function to run on project creation
- Title: title of the template in the Gadget
- OpenFiles: which file to open at launch
- Icon: icon

Project Templates - input widgets

```
Parameter: check
Widget: CheckboxInput
Label: Checkbox Input
Default: On
Position: left
```

- Parameter: The name of the parameter that will be passed to the function that creates the project.
- Widget: The type of the widget (`CheckboxInput` / `SelectInput` / `TextInput` / `FileInput`)
- Label: label to display
- Default: Default value of the element
- Position: where to put this element in the widget

Let's practice !



Now it's your turn to create an addin

Pick an idea (or choose your own)

- Create a template for a data analysis (markdown or project)
- Create a Markdown template with a custom CSS
- Create a template to connect to a database
- Create a template for launching a twitter data scraping