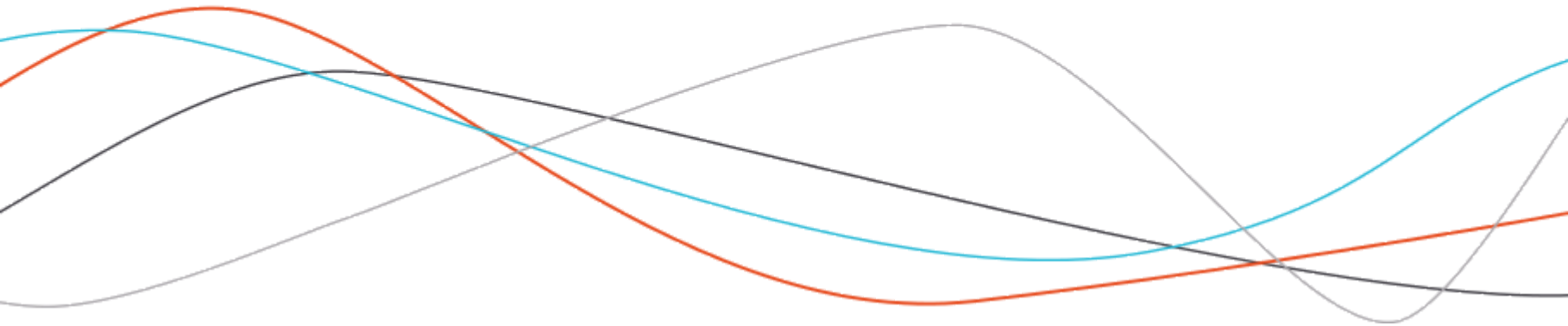
THINKR

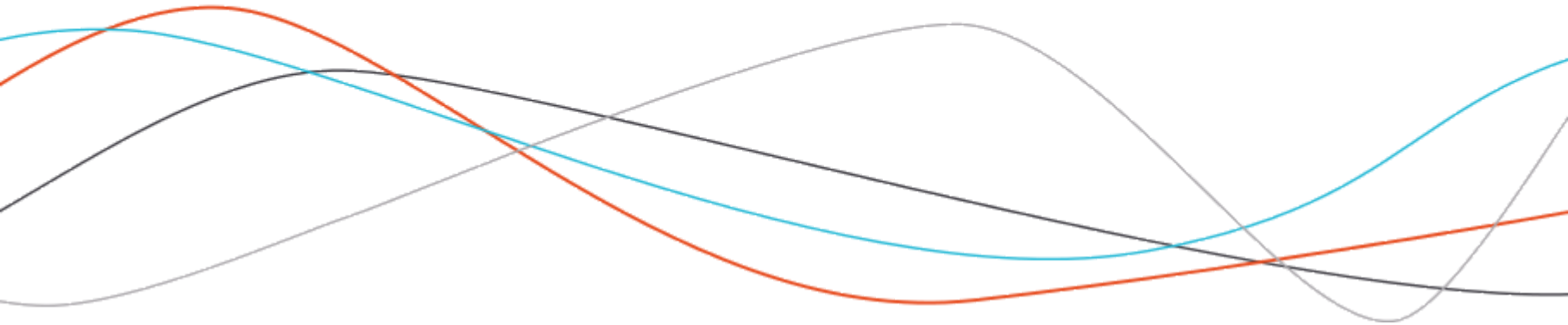
Building Successful Shiny Apps with {golem}

Colin Fay - ThinkR

PART 03 - JavaScript and CSS



```
golem_add_external_resources()
```



golem_add_external_resources()

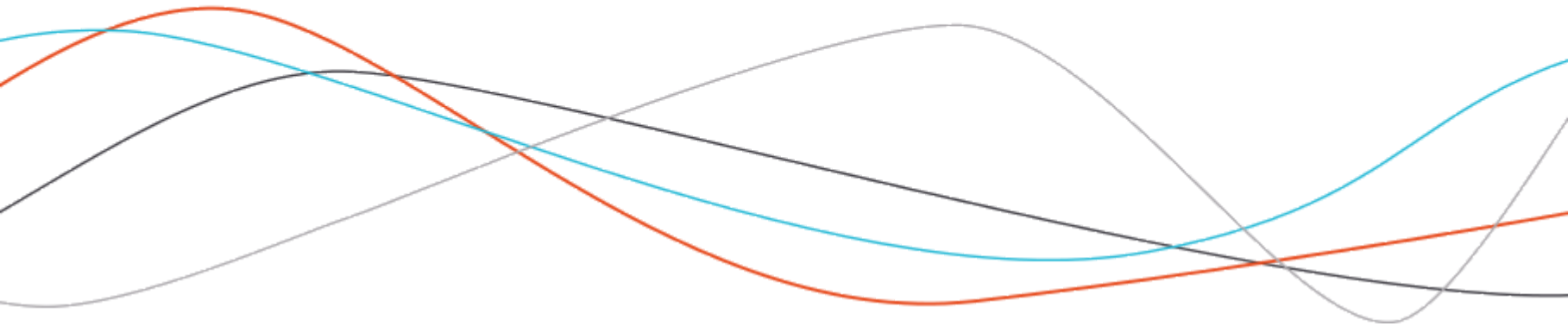
- In `app_ui.R`, the `golem_add_external_resources()` functions add to the app every `.css` and `.js` file contained in `inst/app/www`

```
golem_add_external_resources <- function(){

  add_resource_path(
    'www', app_sys('app/www')
  )

  tags$head(
    favicon(),
    bundle_resources(
      path = app_sys('app/www'),
      app_title = 'golex'
    )
    # Add here other external resources
    # for example, you can add shinyalert::useShinyalert()
  )
}
```

A GENTLE INTRODUCTION TO CSS



What is CSS?

- CSS, (Cascading Style Sheets), is one of main technologies that power the web today, along with HTML and JavaScript
- CSS handles the design, i.e. the visual rendering of the web page: the color of the header, the font, the background, and everything we see

> Example: try the Web Developer extension of Google Chrome, and remove the CSS from a page

- In Shiny, there is a default CSS: the one from Bootstrap 3

Getting started with CSS

- Written in a `.css` file
- CSS syntax is composed of two elements: a selector, and a declaration
- CSS selector: describes how to identify the HTML tags
- Declaration: how is the selected style affected

```
h2 {
  color:red;
}
```

CSS selectors

- name, id, or class

```
<h2 id = "titleone" class = "standard">One</h2>
```

- name == `h2`, write the name as-is: `h2`
- id == `titleone`, prefix the id with `#`: `#titleone`
- class == `standard`, prefix the class with `.`: `.standard`

CSS selectors

- Can be combined

```
div.standard > p {
  color:red;
}
```

> Selects all `<p>` inside a `<div class = "standard">`

CSS pseudo-class

- React to events

```
a:hover {
  color:red;
}
```

CSS properties

- Declaration block: `key: value;`

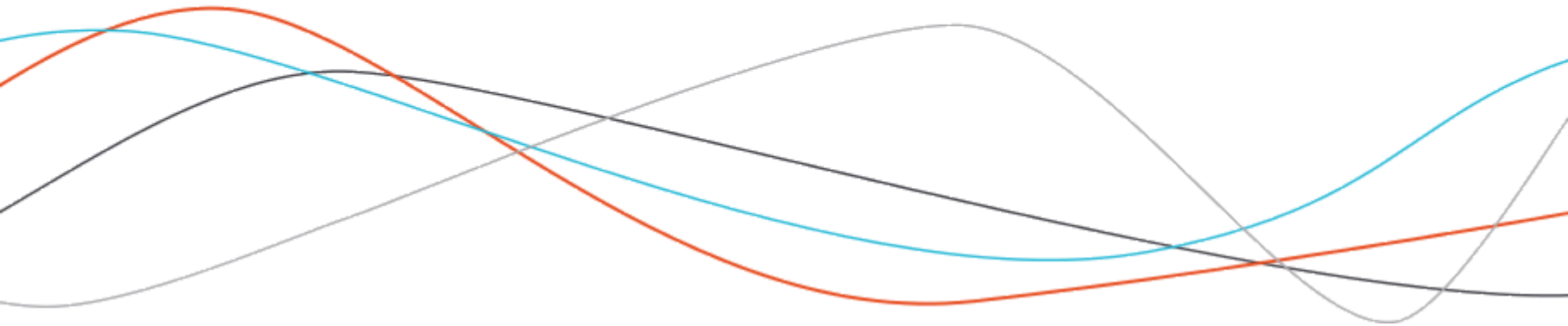
```
text-align: center;
color: red;
```

02_dev.R

- `golem::add_css_file( "custom" )`

```
✓ File created at /Users/colin/golex/inst/app/www/custom.css
✓ File automatically linked in `golem_add_external_resources()`.
• Go to /Users/colin/golex/inst/app/www/custom.css
```

A GENTLE INTRODUCTION TO JAVASCRIPT



What is JavaScript?

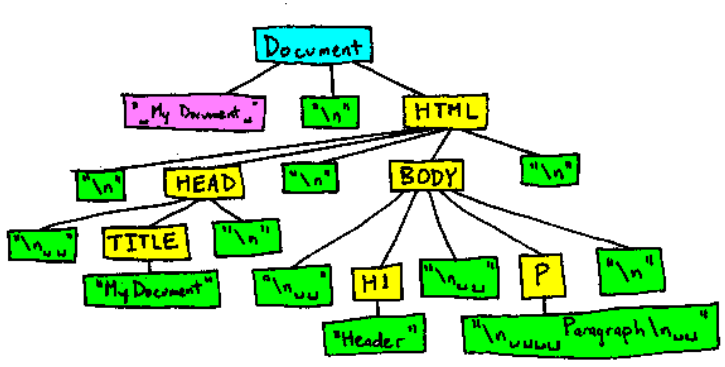
- Scripting language that allows interactivity on webpages
- At the core of Shiny: building a Shiny app is building a JavaScript app that can talk with an R session
- Invisible for most Shiny users
- Built-in in all browser

Why JavaScript?

- Improve the quality of your front-end
- Lower back-end code
- Integrate external librairies

DOM

- A webpage is a DOM (Document Object Model)
- Nodes of a tree where the document is the root
- Contain parents and children
- Nodes contain attributes



Query DOM elements

```
<div id = "pouet" name="plop" class = "plouf">Wesh</div>

document.querySelector("#pouet") // With the ID

document.querySelectorAll(".plouf") // With the class

document.getElementById("pouet") // With the ID

document.getElementsByName("plop") // With the name attribute

document.getElementsByClassName("plouf") // With the class

document.getElementsByTagName("div") // With the tag
```

DOM events

- click / dblclick
- focus
- keypress, keydown, keyup
- mousedown, mouseenter, mouseleave, mousemove, mouseout, mouseover, mouseup
- scroll

> https://developer.mozilla.org/fr/docs/Web/Events

Event Listeners

```
<input type="text" onKeyPress = "alert('plop')">
```

```
<input type="text" id = "plop">
<script>
  document.getElementById("plop").addEventListener("keypress", function(){
    alert("pouet")
  })
</script>
```

In Shiny

- Built-in

```
tags$button(
  "Show",
  onclick = "$('#plot').show()"
)
```

- `tagAppendAttributes()`

```
plotOutput(
  "plot"
) %>% tagAppendAttributes(
  onclick = "alert('hello world')"
)
```

In Shiny

```
library(shiny)
library(magrittr)
ui <- function(request){
  tagList(
    textInput(
      "txt", "Enter txt"
    ) %>% tagAppendAttributes(
      onKeyPress = "alert('plop')"
    )
  )
}

server <- function(input, output, session){

}

shinyApp(ui, server)
```

About jQuery & jQuery selectors

- The jQuery framework is natively included in Shiny

> jQuery makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.

- Very popular JavaScript library which is designed to manipulate the DOM, its events and its elements

> jQuery, the most popular JavaScript library ever created, is used in 85.03% of desktop pages and 83.46% of mobile pages.

https://almanac.httparchive.org/en/2019/javascript#first-party-vs-third-party

About jQuery & jQuery selectors

- jQuery selectors are CSS selectors, plus some custom
- Are used inside `$()`
- Then call a built-in function

```
$("#plop").hide()
```

> Hides the element of class "plop"

Examples

- `$('#id').show();` and `$('#id').hide()`

  > Show and hide

- `$('#id').css("color", "red);`

  > Changes the CSS to red

- `$("#id").text( "this" );`

  > Changes the text content to 'this'

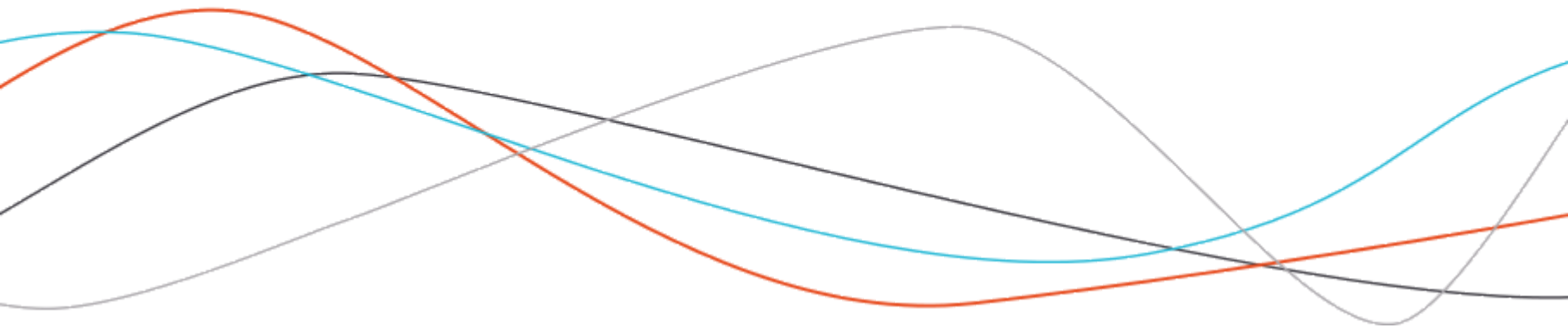
- `$("#id").remove();`

  > Removes element from UI

Add event

```
var x = $("#pouet");
x.on("click", function(){
  $(this).attr("value", parseInt($(this).attr("value")) + 1 )
})
```

JavaScript <-> Shiny communication



From R to JavaScript

- In `inst/app/www/handlers.js`

```
$( document ).ready(function() {
  Shiny.addCustomMessageHandler('fun', function(arg) {

  })
});
```

- Called from R with

```
session$sendCustomMessage("fun", list())
```

- In `inst/app/www/handlers.js`

```
$( document ).ready(function() {
  Shiny.addCustomMessageHandler('computed', function(mess) {
    alert("Computed " + mess.what + " in " + mess.sec + " secs");
  })
});
```

- Called from R with

```
observe({
  deb <- Sys.time()
  Sys.sleep(5)
  session$sendCustomMessage(
    "computed",
    list(
      what = "plop",
      sec = round(Sys.time() - deb)
    )
  )
})
```

From JavaScript to R

- In `inst/app/www/handlers.js`

```
Shiny.setInputValue("rand", Math.random())
```

- Received in R with

```
observeEvent( input$rand , {
  print( input$rand )
})
```

02_dev.R

- golem::add_js_file("script")
- golem::add_js_handler("handlers")

```
golem::add_js_handler( "handlers" )
```

```
$( document ).ready(function() {
  Shiny.addCustomMessageHandler('fun', function(arg) {

  })
});
```

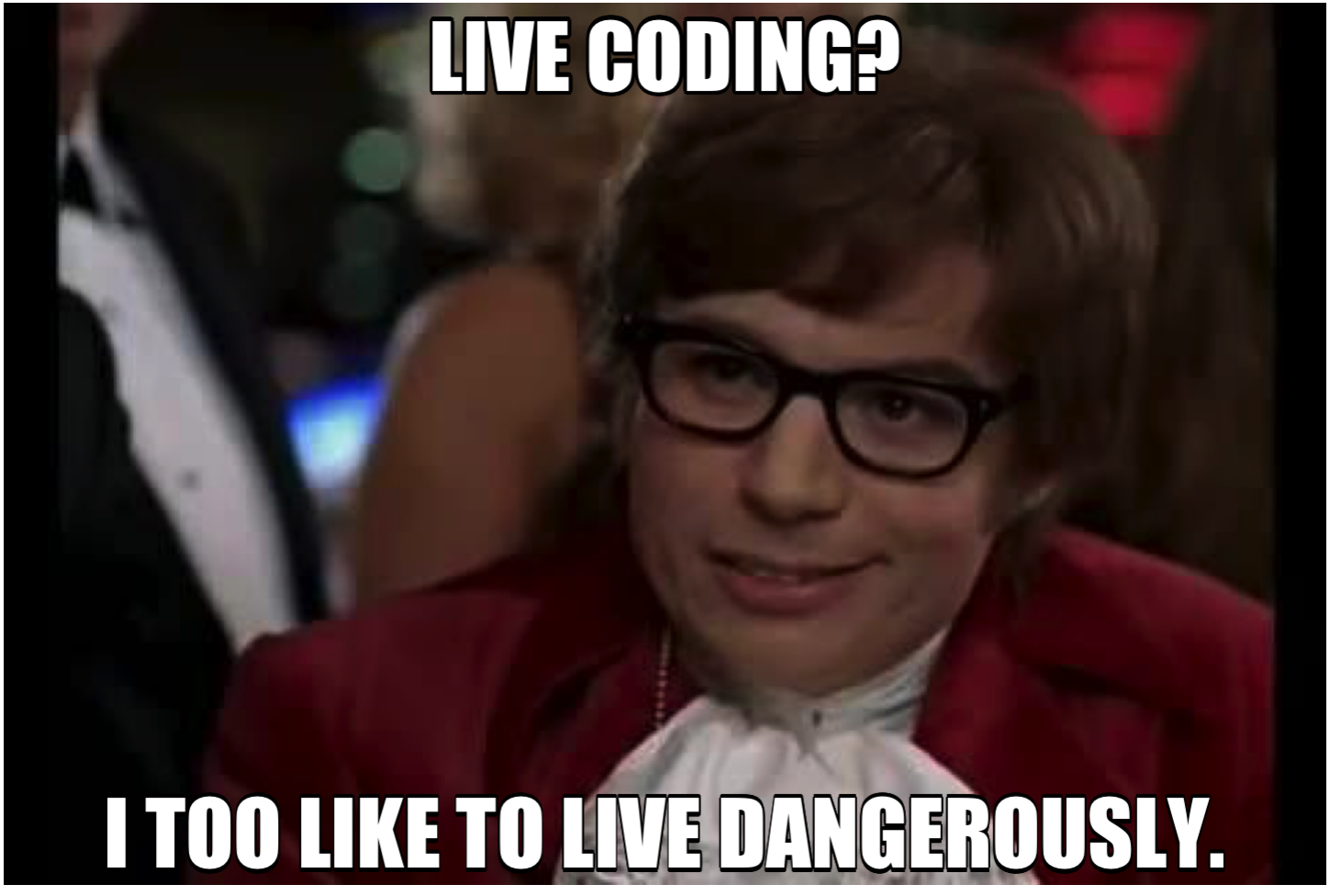
{golem} js functions

- `{golem}` comes with a series of built-in functions
- They can be called with `golem::invoke_js()`

```
golem::invoke_js("showid", ns("plot"))
```

- See `?golem::activate_js`

Demo time 🎉



Your turn!

https://github.com/ColinFay/golem-joburg/tree/master/exo-part-3