# shinyjs

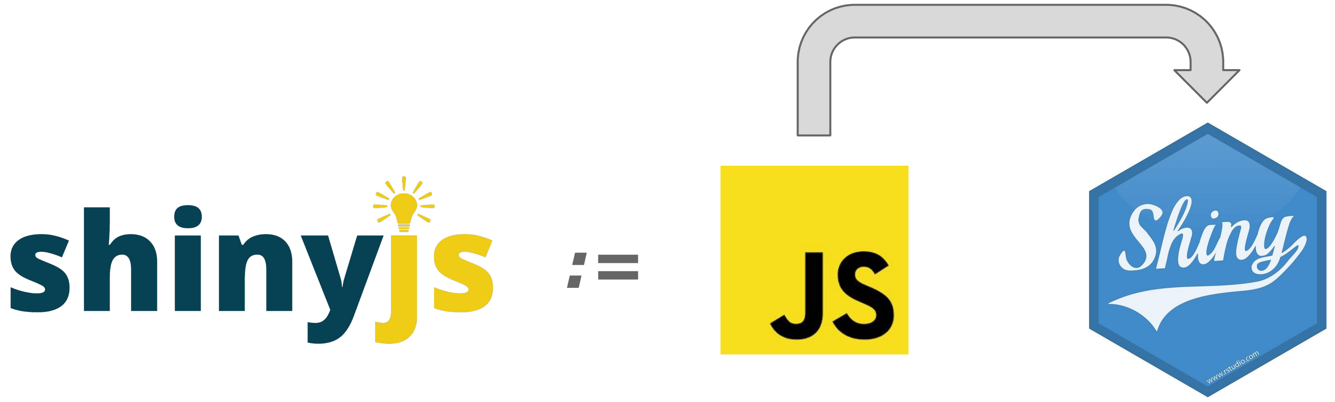
Agata Pałdyna
Rafał Muszyński
Tymoteusz Makowski

## Agenda

1) Czym jest shinyjs?
2) Co umożliwia?
3) Czego wymaga?
4) Przykłady
5) Cheatsheet

## Czym jest shinyjs?

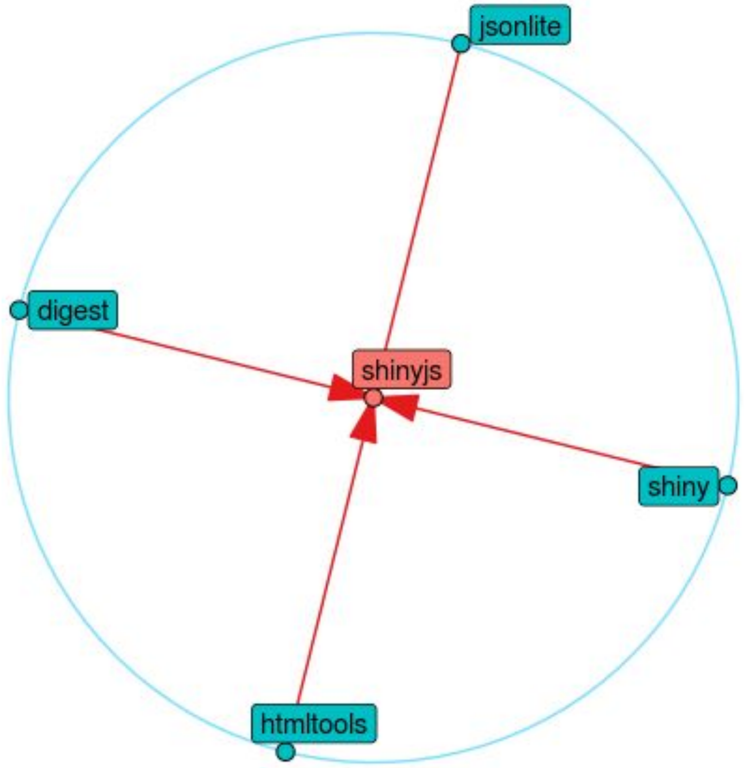
- Pakiet języka R
- Umożliwia wykonywanie operacji JavaScript w aplikacjach Shiny
- **Nie trzeba znać JavaScript!**

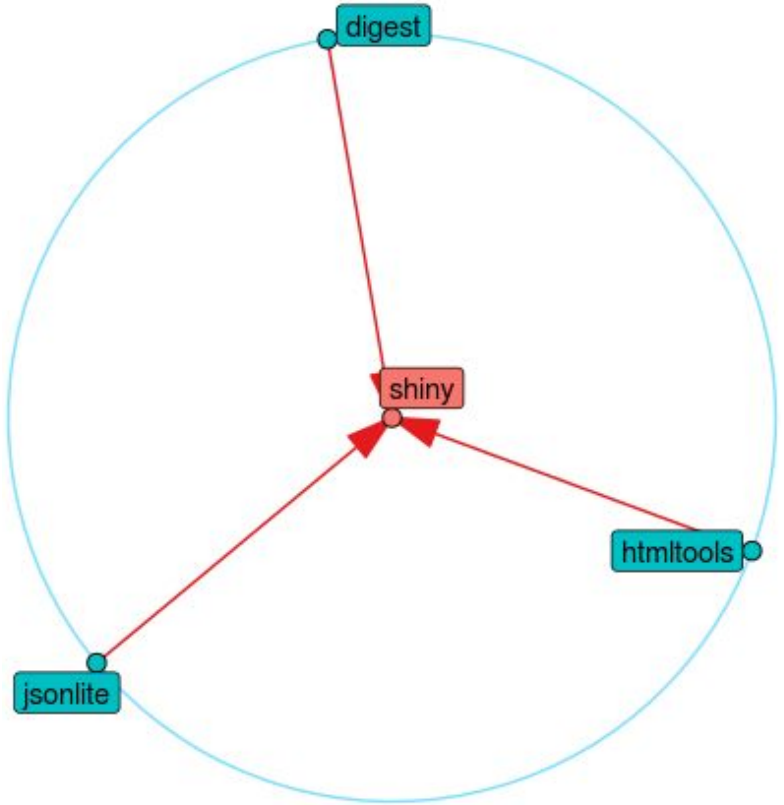


## Co umożliwia?

- Ukrywać/pokazywać elementy
- Wyłączyć/włączyć input
- Zresetować input do pierwotnej wartości
- Opóźnić wykonywanie kodu
- Łatwo wywoływać własne funkcje js z poziomu R

## Czego wymaga?

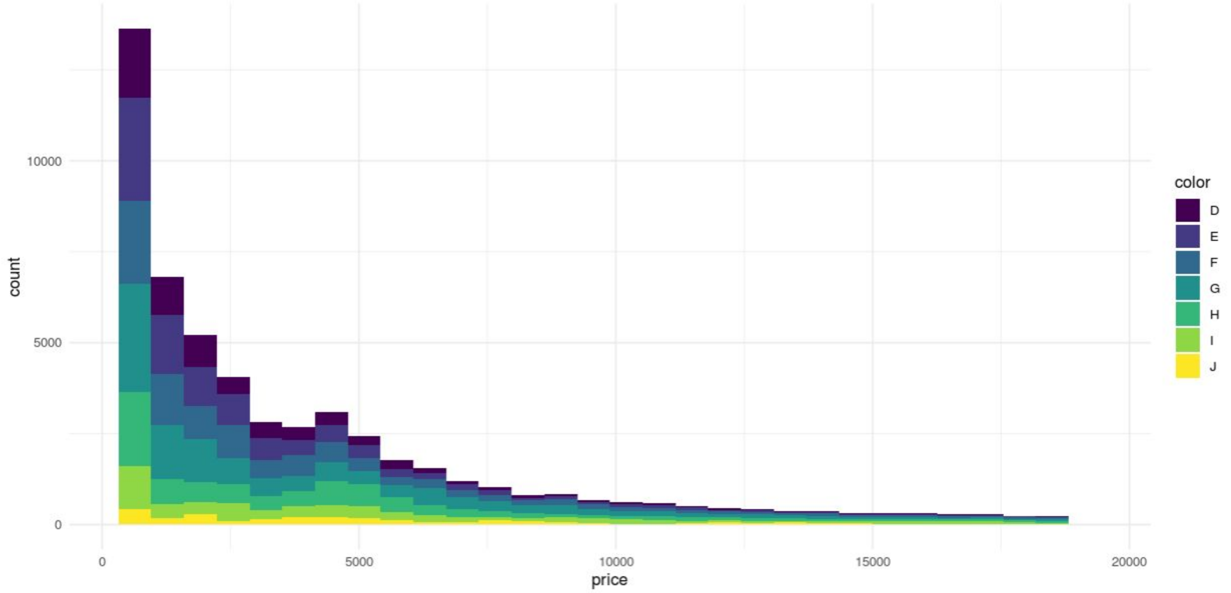




## Przykłady

# shinyJS example

Do a thing



```r
library("shiny"); library("shinipsum")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        plotOutput("plot")
    )
)

server <- function(input, output) {
    output[["plot"]] <- renderPlot(random_ggplot())
}

shinyApp(ui = ui, server = server)
```

```r
library("shiny"); library("shinipsum")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        plotOutput("plot")
    )
)

server <- function(input, output) {
    output[["plot"]] <- renderPlot(random_ggplot())
}

shinyApp(ui = ui, server = server)
```

```r
library("shiny"); library("shinipsum")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        uiOutput("ui")
    )
)

server <- function(input, output) {
    rv <- reactiveValues(hidden = FALSE)

    observeEvent(input[["button"]], {
        rv_val <- isolate(rv$hidden)
        rv$hidden <- !rv_val
    })

    output[["ui"]] <- renderUI({
        if (!rv$hidden) {
            plotOutput("plot")
        }
    })

    output[["plot"]] <- renderPlot(random_ggplot())
}
```

```r
library("shiny"); library("shinipsum")
library("shinyjs")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        useShinyjs(),
        plotOutput("plot")
    )
)

server <- function(input, output) {
    observeEvent(input[["button"]], toggle("plot"))
    output[["plot"]] <- renderPlot(random_ggplot())
}
```

```r
library("shiny"); library("shinipsum")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        uiOutput("ui")
    )
)

server <- function(input, output) {
    rv <- reactiveValues(hidden = FALSE)

    observeEvent(input[["button"]], {
        rv_val <- isolate(rv$hidden)
        rv$hidden <- !rv_val
    })

    output[["ui"]] <- renderUI({
        if (!rv$hidden) {
            plotOutput("plot")
        }
    })

    output[["plot"]] <- renderPlot(random_ggplot())
}
```

```r
library("shiny"); library("shinipsum")
library("shinyjs")

ui <- pageWithSidebar(
    headerPanel("shinyJS example"),
    sidebarPanel(
        actionButton("button", "Do a thing")
    ),
    mainPanel(
        useShinyjs(),
        plotOutput("plot")
    )
)

server <- function(input, output) {
    observeEvent(input[["button"]], toggle("plot"))
    output[["plot"]] <- renderPlot(random_ggplot())
}
```

| **Wariant** | **Czas implementacji** |
|---|---|
| Z shinyJS | 1:55 |
| Bez shinyJS | 8:25 |

## shinyjs, czyli pisanie po angielsku

```r
library("shiny")
library("shinipsum")
library("shinyjs")


ui <- fluidPage(
    useShinyjs(),
    h1("A title."),
    actionButton("button", "A button.")
)


server <- function(input, output) {
    onclick("button", runjs('alert("A notification from Shiny!")'))
}


shinyApp(ui = ui, server = server)
```

# shinyjs, czyli pisanie po angielsku

```
library("shiny")
library("shinipsum")
library("shinyjs")


ui <- fluidPage(
    useShinyjs(),
    h1("A title."),
    actionButton("button", "A button.")
)


server <- function(input, output) {
    onclick("button", runjs('alert("A notification from Shiny!")'))
}


shinyApp(ui = ui, server = server)
```

## Cheatsheet

# shinyjs CHEAT SHEET

Perform common useful JavaScript operations in Shiny apps without any JS background

## Before you start

**useShinyjs(**rmd, debug, html**)**
*In order to use any shinyjs function in a Shiny app, you must first call useShinyjs() anywhere in the app's UI*

**rmd**
Enable to use inside an interactive R markdowndocument
**debug**
Enable default debugging in JS console
**html**
Enable if your shiny app builds the entire user interface with a custom HTML file.

## Interactive mode

**runcode**
*Adds a text input to your app that lets you run arbitrary R code live. To enable, call runcodeUI() in the UI and runcodeServer() in serverFunction*

## Logs

**showLog()**
*Print any JavaScript console.log() messages in the R console*

**logjs(**text**)**
*Print a message to the JavaScript console*

## Events

**onevent(**event, id, expr, add**)**
*Run an R expression when an element is clicked. Use onclick() when handling mouse clicks and skip the event parameter*

**expr**
R expression or function to run after the event is triggered
**add**
Enable if expr should run after previously defined onevent calls. Otherwise they are overwritten

## CSS

**addClass / removeClass / toggleClass**
**(**id, class, selector, condition**)**
*Adds or removes class from CSS element*

**class**
The CSS class to add/remove

**inlineCSS(**rules**)**
*Easily add inline CSS to a Shiny app.*

**rules**
string with valid CSS code or list(selector = declarations) where declarations is a string or vector of declarations.

**https://deanattali.com/shinyjs**

# shinyjs CHEAT SHEET

Perform common useful JavaScript operations in Shiny apps without any JS background

## State functions

**enable / disable / toggleState (**id, selector, condition**)**
*Enable or disable an input element, such as a button or a text input*

**disabled(**...**)**
*Initialize Shiny input as disabled*

**...**
Shiny input (or tagList or list of tags) to disable

**reset(**id**)**
*Reset input widget to it's original state.*

**id**
The id of the input element or the id of an HTML tag with input elements

## Custom JavaScript

**runjs(**code**)**
*Run arbitrary JS code*

**extendShinyJS(**script, text, functions**)**
*Write your own JS functions and run them with shinyjs()*

## Visibility

**show / show Element / hide / toggle / toggleElement (**id, anim, animType, time, selector**)**
*Display element. Use showElement and toggleElement for S4 objects.*

**anim**
if TRUE then animate the behaviour
**animType**
The type of animation, "slide" or "fade"
**time**
Animation length in seconds.

**hidden(**...**)**
*Initialize a Shiny tag as invisible*

**...**
Shiny input (or tagList or list of tags) to make invisible

## Other

**html(**id, html, add, selector**)**
*Change HTML of an element*

**html**
HTML/text to place inside element
**add**
if TRUE then append html to the existing contents

**alert / info (**text**)**
*show message to the user*

**delay(**ms, expr**)**
*Execute R code with a delay*

**ms**
delay length in miliseconds

**Common Parameters**

**id**
The id of the element/Shiny tag
**selector**
JQuery selector. Ignored if the id argument is given
**condition**
When is the toggle action performed

**https://deanattali.com/shinyjs**

## Dziękujemy za uwagę!

Bibliografia:

- https://deanattali.com/shinyjs/
- https://github.com/daattali/shinyjs