



Designing UI

Daniel Kaplan

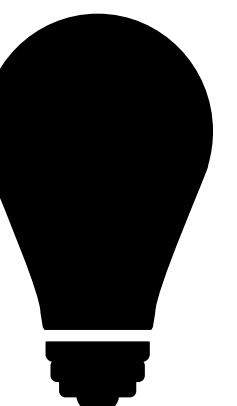
dtkaplan 
dtkaplan@gmail.com 

- Web application UI is ultimately HTML/CSS/JavaScript
- Let's R users write user interfaces using a simple, familiar-looking API...
- ...but no limits for advanced users



Tips

- It's tempting to create a beautifully laid out app from the beginning.
- Resist the temptation.
 - Make a prototype:
 - Decide on names/types of the various controls
 - Write observers and renderers, keeping it simple at first and then defining reactive functions and reactive values as they make sense.
- Use .Rmd instead of .R



Interface builder functions



tags

```
> names(tags)
```

```
[1] "a"      "abbr"    "address" "area"    "article"
[6] "aside"  "audio"   "b"       "base"    "bdi"
[11] "bdo"    "blockquote" "body"    "br"      "button"
[16] "canvas" "caption" "cite"    "code"    "col"
[21] "colgroup" "command" "data"    "datalist" "dd"
[26] "del"    "details" "dfn"     "div"     "dl"
[31] "dt"     "em"      "embed"   "eventsource" "fieldset"
[36] "figcaption" "figure"  "footer"  "form"     "h1"
[41] "h2"     "h3"     "h4"     "h5"     "h6"
[46] "head"    "header"
[51] "i"      "iframe"
[56] "kbd"    "keygen"
[61] "link"    "mark"    "map"     "menu"    "meta"
[66] "meter"  "nav"     "noscript" "object"  "ol"
[71] "optgroup" "option"  "output"  "p"       "param"
[76] "pre"    "progress" "q"       "ruby"    "rp"
[81] "rt"     "s"       "samp"    "script"  "section"
[86] "select" "small"   "source"  "span"    "strong"
[91] "style"  "sub"     "summary" "sup"     "table"
[96] "tbody"  "td"     "textarea" "tfoot"   "th"
[101] "thead"  "time"   "title"   "tr"     "track"
[106] "u"      "ul"     "var"     "video"   "wbr"
```

<i> some text </i>

tag → HTML

```
> tags$b("This is my first app")
```

```
<b>This is my first app</b>
```



Header tags

```
library(shiny)

# Define UI with tags
ui <- fluidPage(
  tags$h1("First level heading"),
  tags$h2("Second level heading"),
  tags$h3("Third level heading")
)

# Define server fn that does nothing :)
server <- function(input, output) {}

# Create the app object
shinyApp(ui = ui, server = server)
```



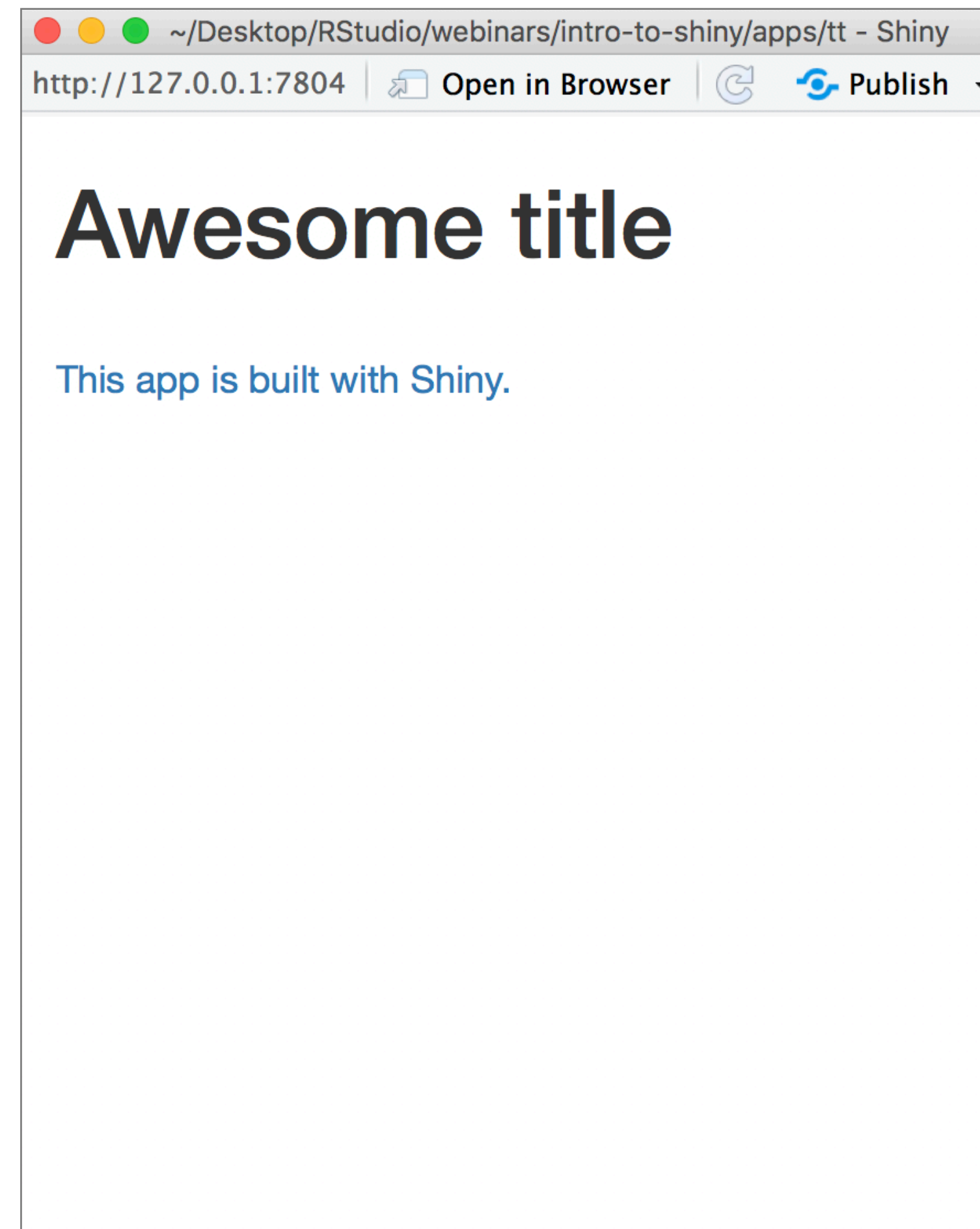
Linked text

```
library(shiny)

# Define UI with tags
ui <- fluidPage(
  tags$h1("Awesome title"),
  tags$br(), # line break
  tags$a("This app is built with Shiny.", href = "http://shiny.rstudio.com/")
)

# Define server fn that does nothing :)
server <- function(input, output) {}

# Create the app object
shinyApp(ui = ui, server = server)
```



Nested tags

```
library(shiny)

# Define UI with tags
ui <- fluidPage(
  tags$p("Lorem ipsum",
    tags$em("dolor"), "sit amet,",
    tags$b("consectetur"),
    "adipiscing elit.")
)

# Define server fn that does nothing :)
server <- function(input, output) {}

# Create the app object
shinyApp(ui = ui, server = server)
```



tags\$p(...)		p(...)
tags\$h1(...)		h1(...)
tags\$h2(...)		h2(...)
tags\$h3(...)		h3(...)
tags\$h4(...)		h4(...)
tags\$h5(...)		h5(...)
tags\$h6(...)		h6(...)
tags\$a(...)		a(...)
tags\$br(...)		br(...)
tags\$div(...)		div(...)
tags\$span(...)		span(...)
tags\$pre(...)		pre(...)
tags\$code(...)		code(...)
tags\$img(...)		img(...)
tags\$strong(...)		strong(...)
tags\$em(...)		em(...)
tags\$hr(...)		hr(...)



Common tags

Commonly used tags have wrappers with short names.

All of these are just wrappers on tag().

Common tags

```
> tags$a("Anchor text")
```

```
<a>Anchor text</a>
```

```
> a("Anchor text")
```

```
<a>Anchor text</a>
```

```
> tags$br()
```

```
<br/>
```

```
> br()
```

```
<br/>
```

```
> tags$code("Monospace text")
```

```
<code>Monospace text</code>
```

```
> code("Monospace text")
```

```
<code>Monospace text</code>
```

```
> tags$h1("First level header")
```

```
<h1>First level header</h1>
```

```
> h1("First level header")
```

```
<h1>First level header</h1>
```



HTML

```
> HTML("Hello world, <br/> and then a line break.")
```

```
Hello world, <br/> and then a line break.
```



Your turn

- Start with `movies_11.R`.
- Add some helper text to the app using tags that let your users know how to navigate the app.
 - For instance, insert some HTML above a control.
- **Stretch goal:** Arrange control labels to have pop-up text explaining them.



5_m 00_s

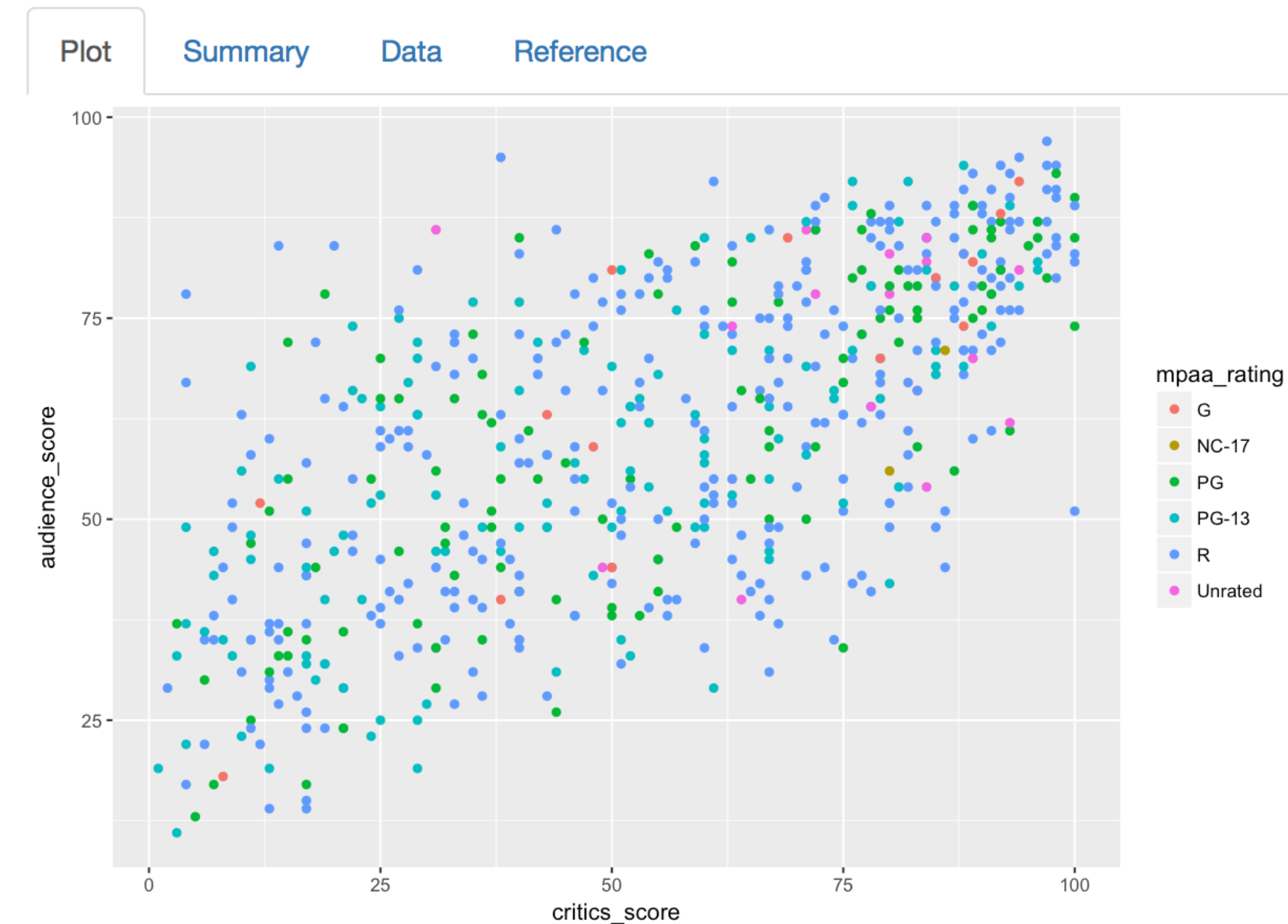
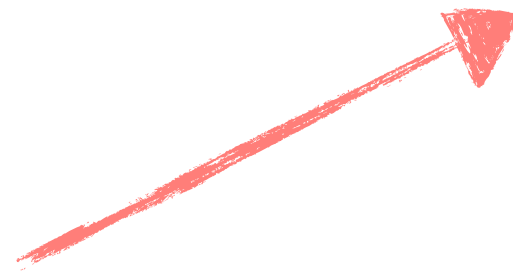


Tabs



tabPanel()

```
mainPanel(  
  tabsetPanel(type = "tabs",  
    tabPanel("Plot", plotOutput("plot")),  
    tabPanel("Summary", tableOutput("summary")),  
    tabPanel("Data", DT::dataTableOutput("data")),  
    tabPanel("Reference",  
      tags$p("There data were obtained from",  
        tags$a("IMDB", href = "http://www.imdb.com/"), "and",  
        tags$a("Rotten Tomatoes", href = "https://www.rottentomatoes.com/"), "."),  
      tags$p("The data represent", nrow(movies), "randomly sampled movies released be  
1972 to 2014 in the United States.")  
    )  
  )  
)
```



tabPanel()

```
mainPanel(  
  tabsetPanel(type = "tabs",  
    tabPanel("Plot", plotOutput("plot")),  
    tabPanel("Summary", tableOutput("summary")),  
    tabPanel("Data", DT::dataTableOutput("data")),  
    tabPanel("Reference",  
      tags$p("There data were obtained from",  
        tags$a("IMDB", href = "http://www.imdb.com/"), "and",  
        tags$a("Rotten Tomatoes", href = "https://www.rottentomatoes.com/"), "."),  
      tags$p("The data represent", nrow(movies), "randomly sampled movies released between  
1972 to 2014 in the United States.")  
    )  
  )  
)
```

Plot

Summary

Data

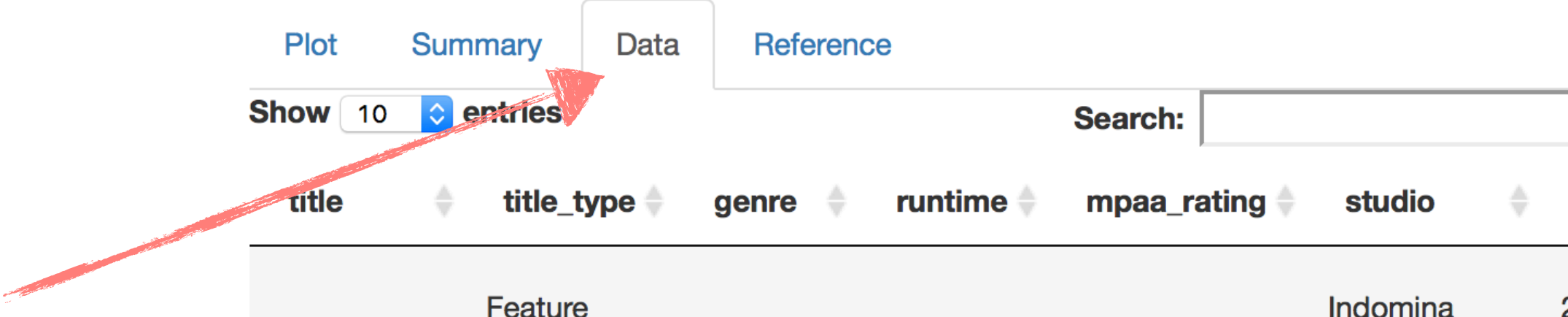
Reference

mpaa_rating	mean_as	sd_as	mean_cs	sd_cs	n	cor
G	66.625	20.656	62.250	27.939	16	0.836
NC-17	63.500	10.607	83.000	4.243	2	1.000
PG	60.418	20.110	54.491	28.503	110	0.733
PG-13	56.015	19.002	46.085	26.518	130	0.662
R	61.454	19.986	56.877	27.463	317	0.648
Unrated	70.812	14.725	74.938	16.631	16	0.105



tabPanel()

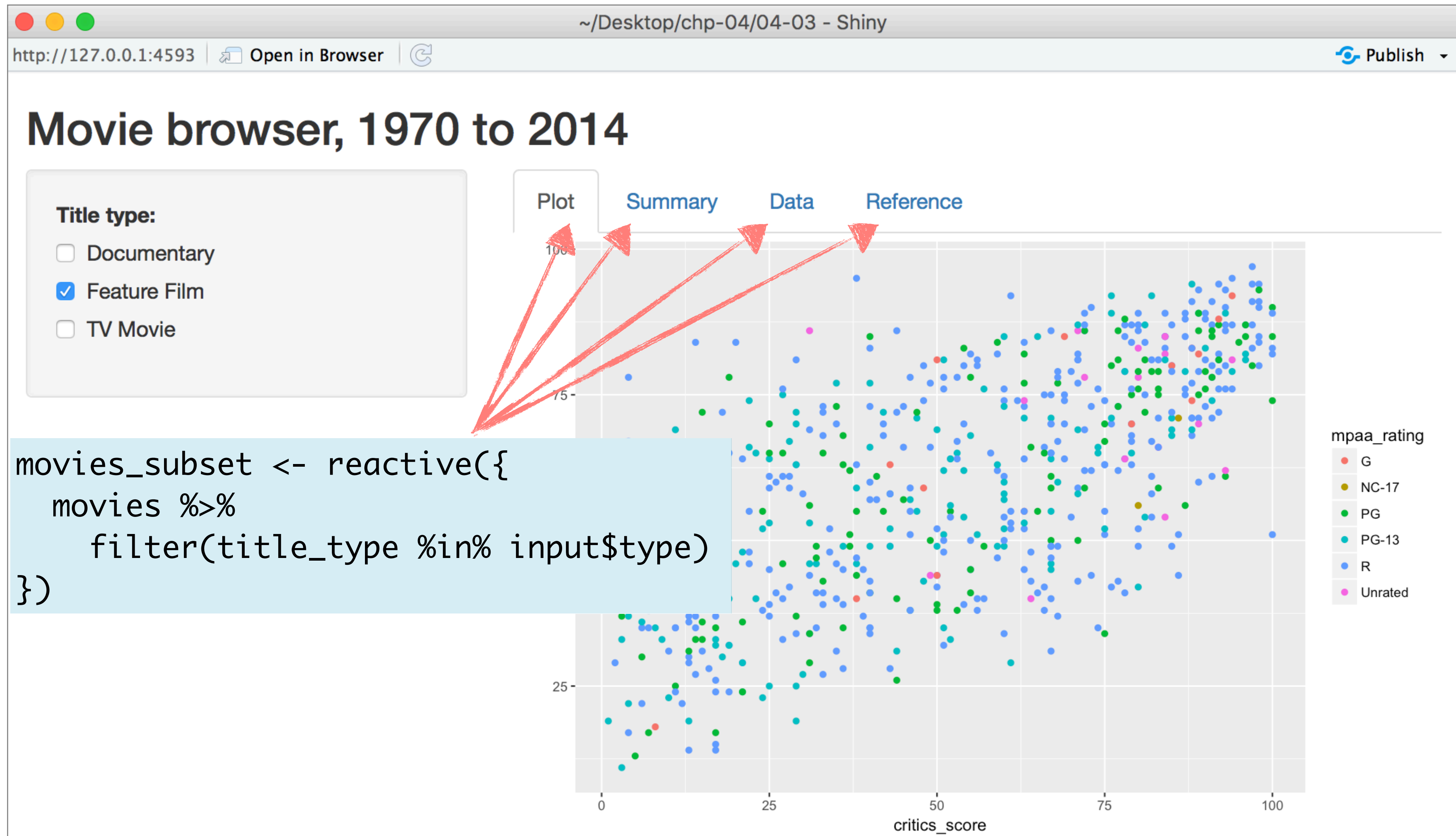
```
mainPanel(  
  tabsetPanel(type = "tabs",  
    tabPanel("Plot", plotOutput("plot")),  
    tabPanel("Summary", tableOutput("summary")),  
    tabPanel("Data", DT::dataTableOutput("data")),  
    tabPanel("Reference",  
      tags$p("There data were obtained from",  
        tags$a("IMDB", href = "http://www.imdb.com/"), "and",  
        tags$a("Rotten Tomatoes", href = "https://www.rottentomatoes.com/"), "."),  
      tags$p("The data represent", nrow(movies), "randomly sampled movies released between  
1972 to 2014 in the United States.")  
    )  
  )  
)
```



Plot Summary Data Reference						
Show 10 entries		Search:				
title	title_type	genre	runtime	mpaa_rating	studio	t
Filly Brown	Feature Film	Drama	80	R	Indomina Media Inc.	2019
The Dish	Feature Film	Drama	101	PG-13	Warner Bros. Pictures	2014
Waiting for Guffman	Feature Film	Comedy	84	R	Sony Pictures Classics	1991
The Age of Innocence	Feature Film	Drama	139	PG	Columbia Pictures	1993
Malevolence	Feature Film	Horror	90	R	Anchor Bay Entertainment	2001
...	Feature	-	Paramount	19...

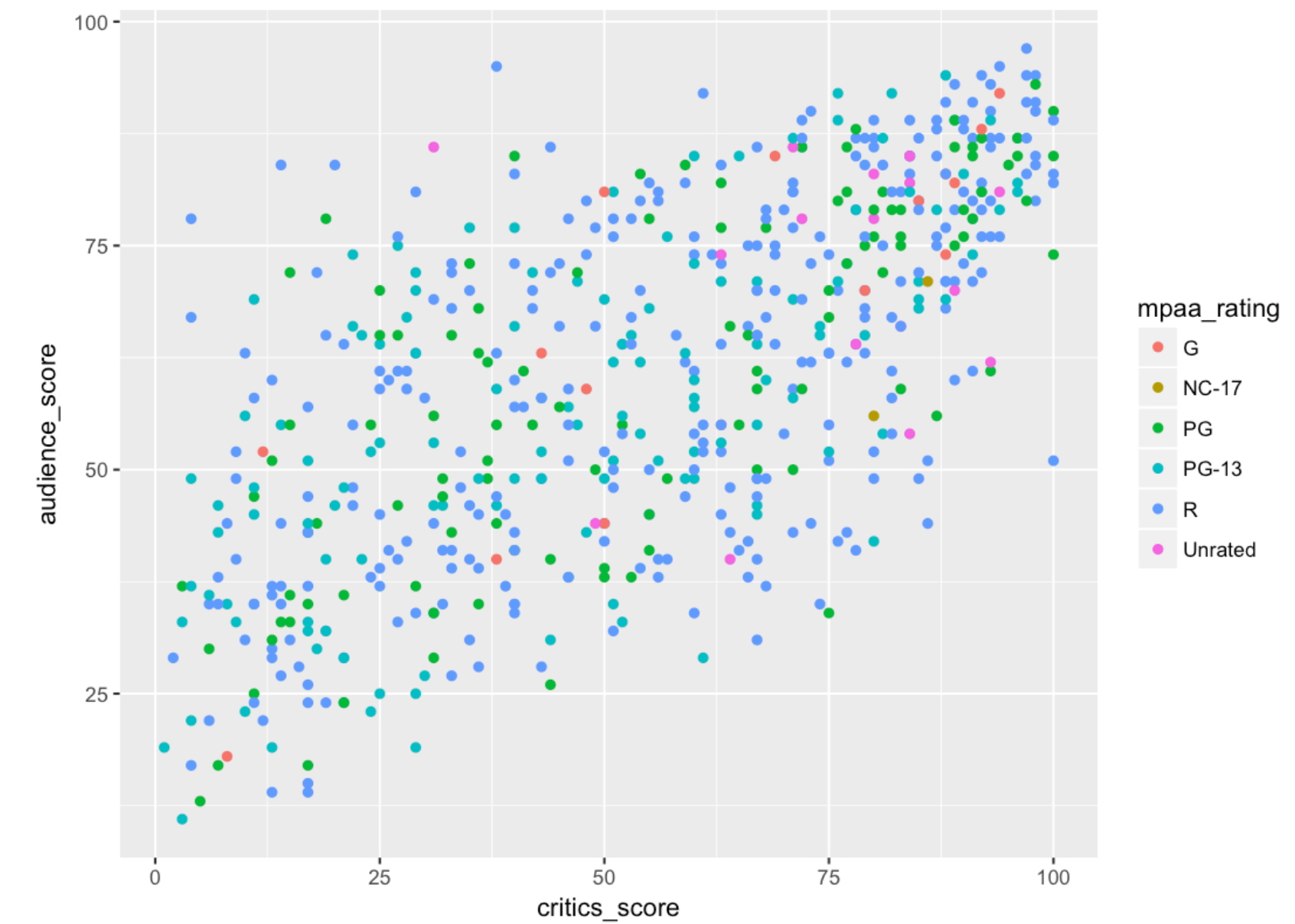


Tabs and reactivity



navlistPanel()

```
mainPanel(  
  navlistPanel(tabPanel("Plot", plotOutput("plot")),  
    tabPanel("Summary", tableOutput("summary")),  
    tabPanel("Data", DT::dataTableOutput("data")),  
    tabPanel("Reference",  
      tags$p("There data were obtained from",  
        tags$a("IMDB", href = "http://www.imdb.com/"), "and",  
        tags$a("Rotten Tomatoes", href = "https://www.rottentomatoes.com/"), ".")  
      tags$p("The data represent", nrow(movies), "randomly sampled movies rel  
between 1972 to 2014 in the United States.")  
    )  
  )  
)
```



Your turn

- Continue working on movies_11.R.
- Split the app into two tabs: one for plot and the other for data table.
- **Stretch goal:** Add another tab for summary statistics and references.

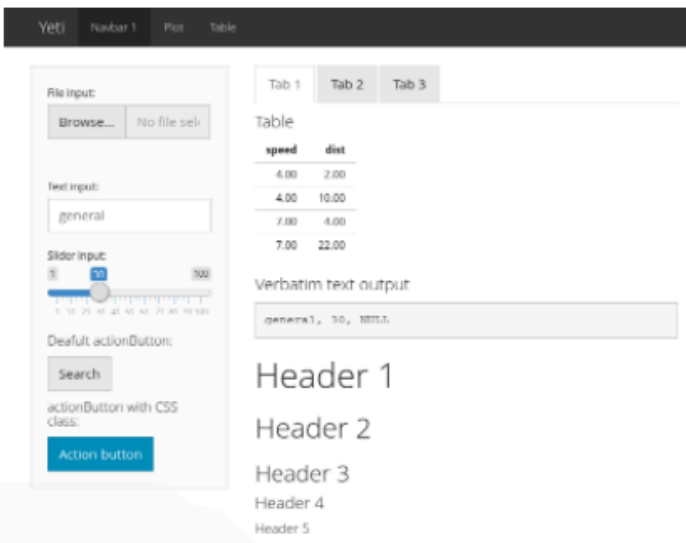
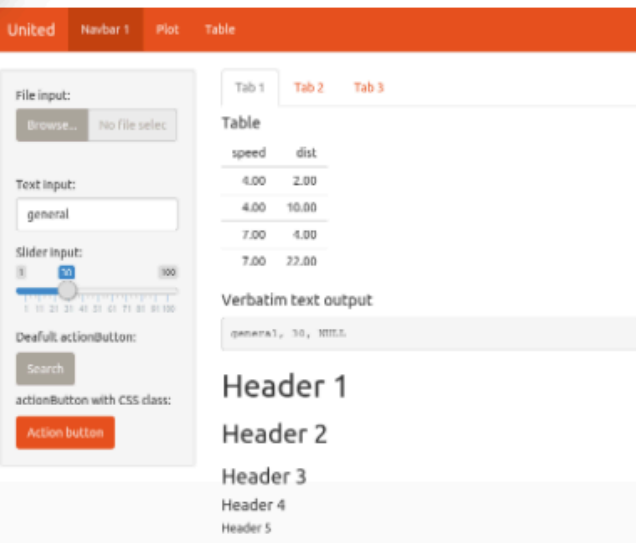
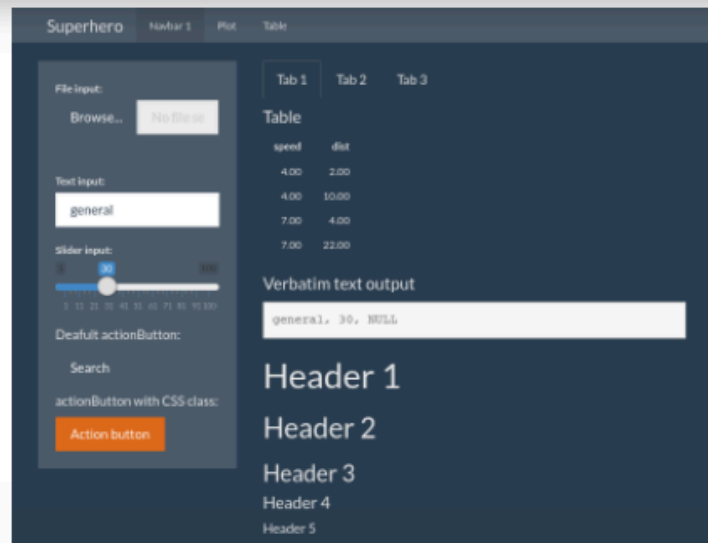
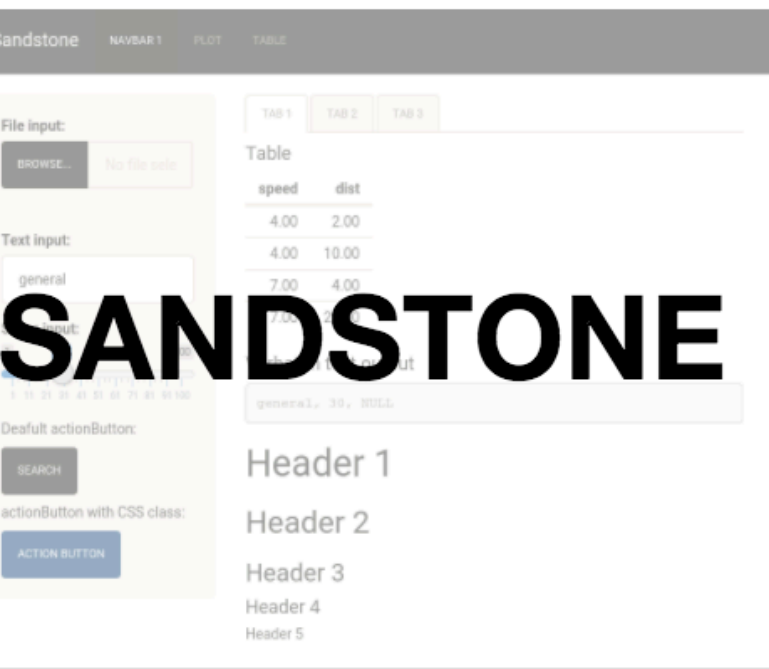
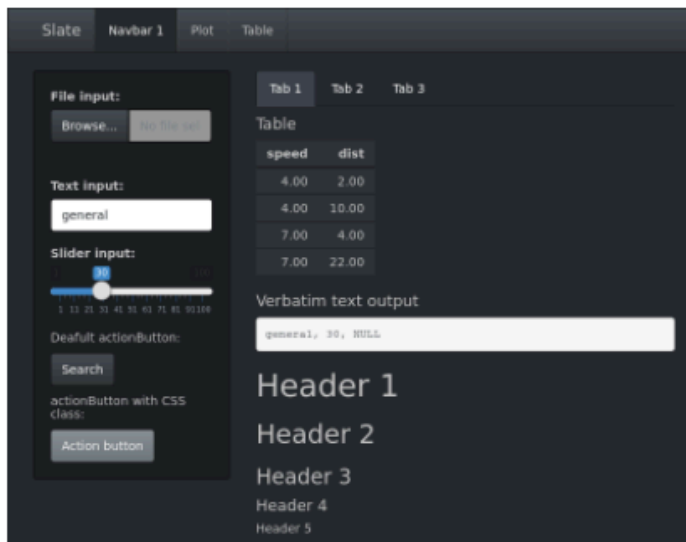
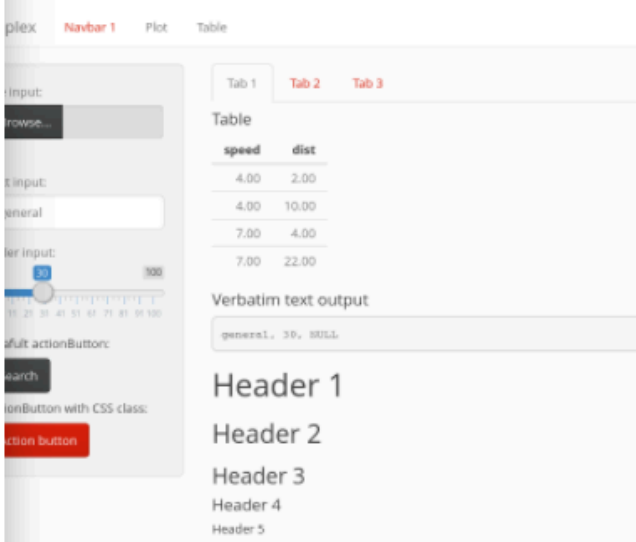
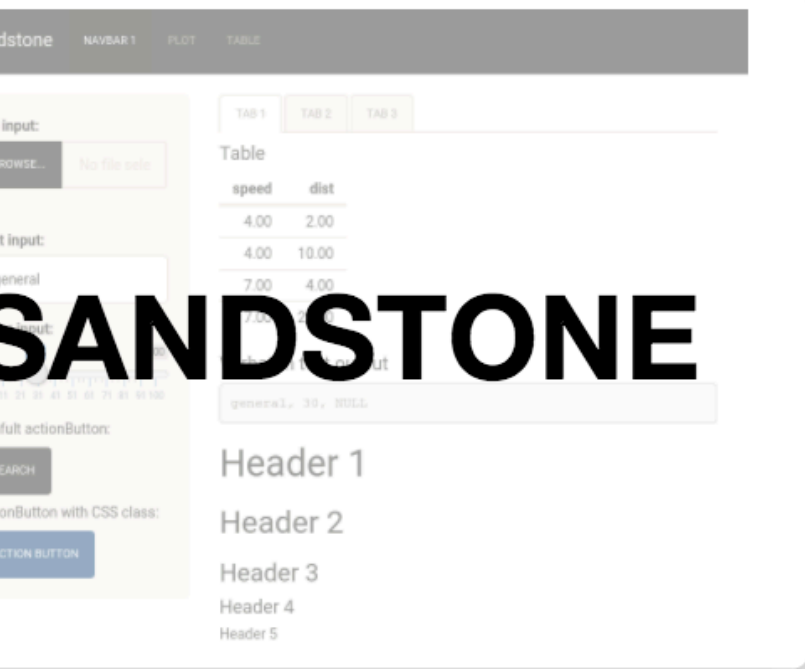
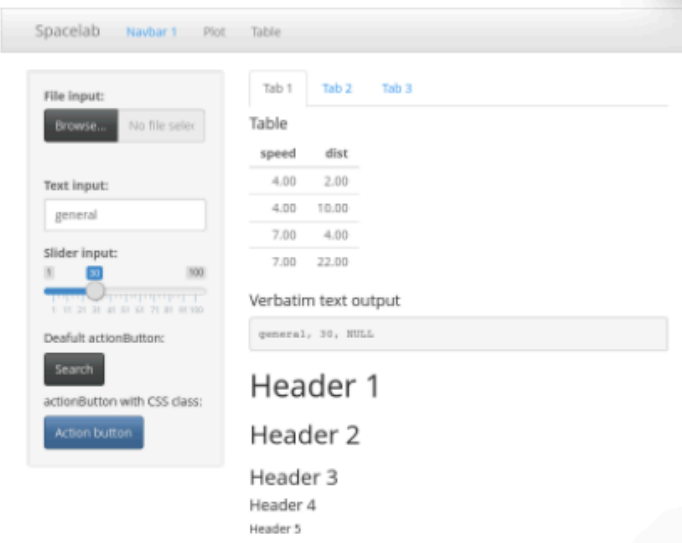


10_m 00_s



shinythemes

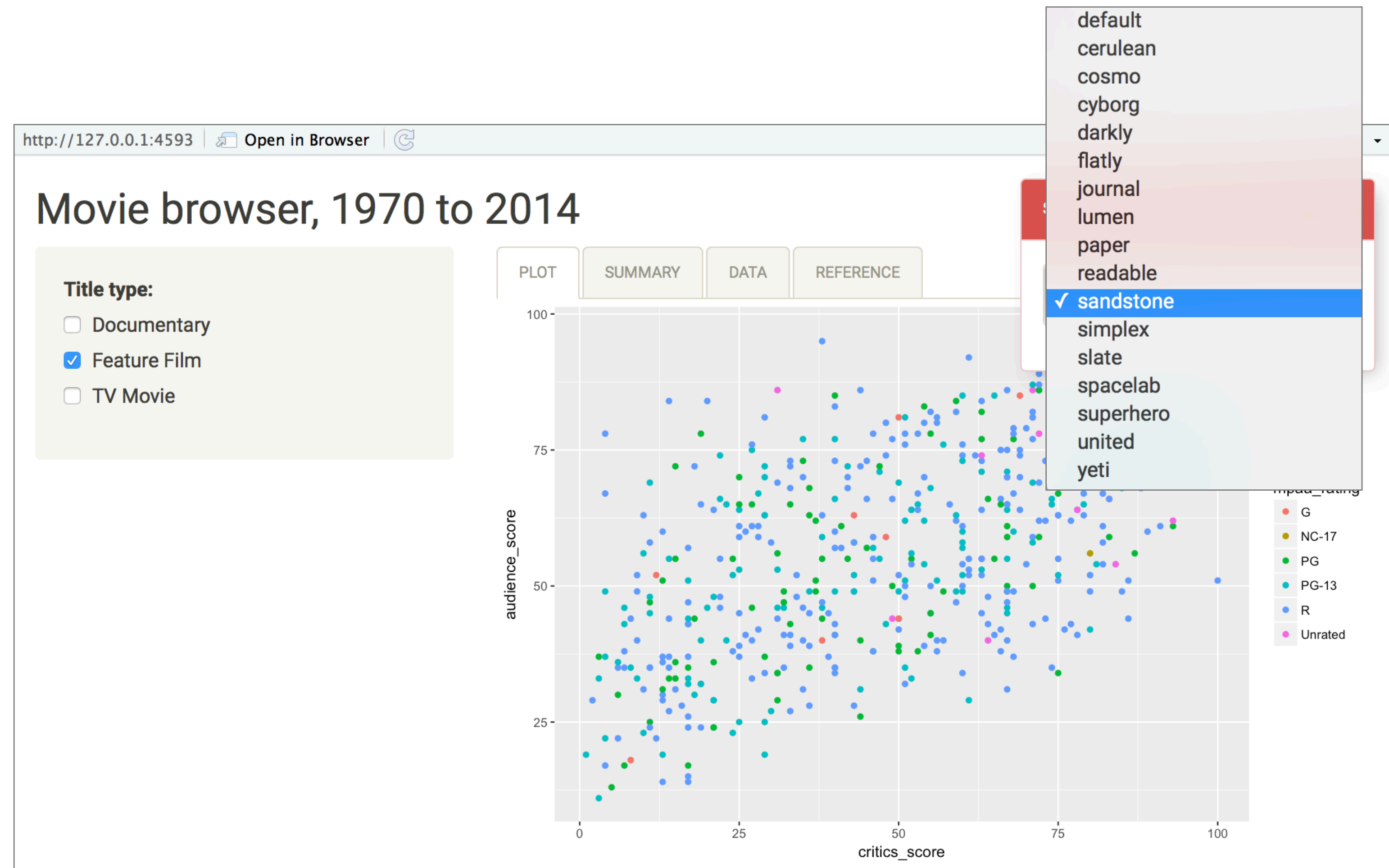




```
library(shiny)
library(shinythemes)
```

```
ui <- fluidPage(
  themeSelector(),
  ...
)
```

shinythemes



Your turn

- Continue working on movies_11.R.
- Add the theme selector, browse various themes, and pick a theme and apply it.
 - Don't forget to remove the selector once you're done picking a theme.
- Add an theme= argument to fluidPage() to implement the theme.



5_m 00_s

