

JS + R == Amazing Applications

Understanding HTMLWidgets

Maya Gans

maya.gans@atorusresearch.com
bit.ly/atoruswidgets



HTMLWidgets

Leaflet

WHAT
ARE
THEY?

WHY
SHOULD I
CARE?

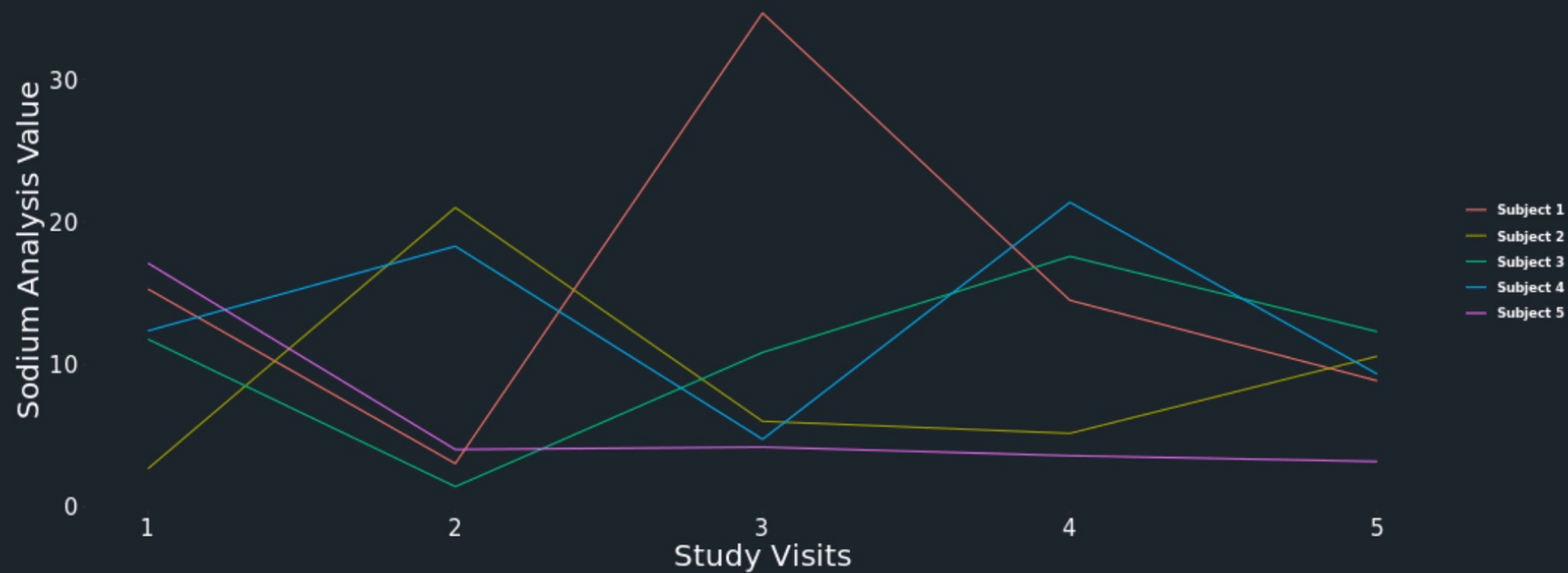
HOW DO I
MAKE
ONE?


plotly

Shiny: An Interlude

A Shiny Application

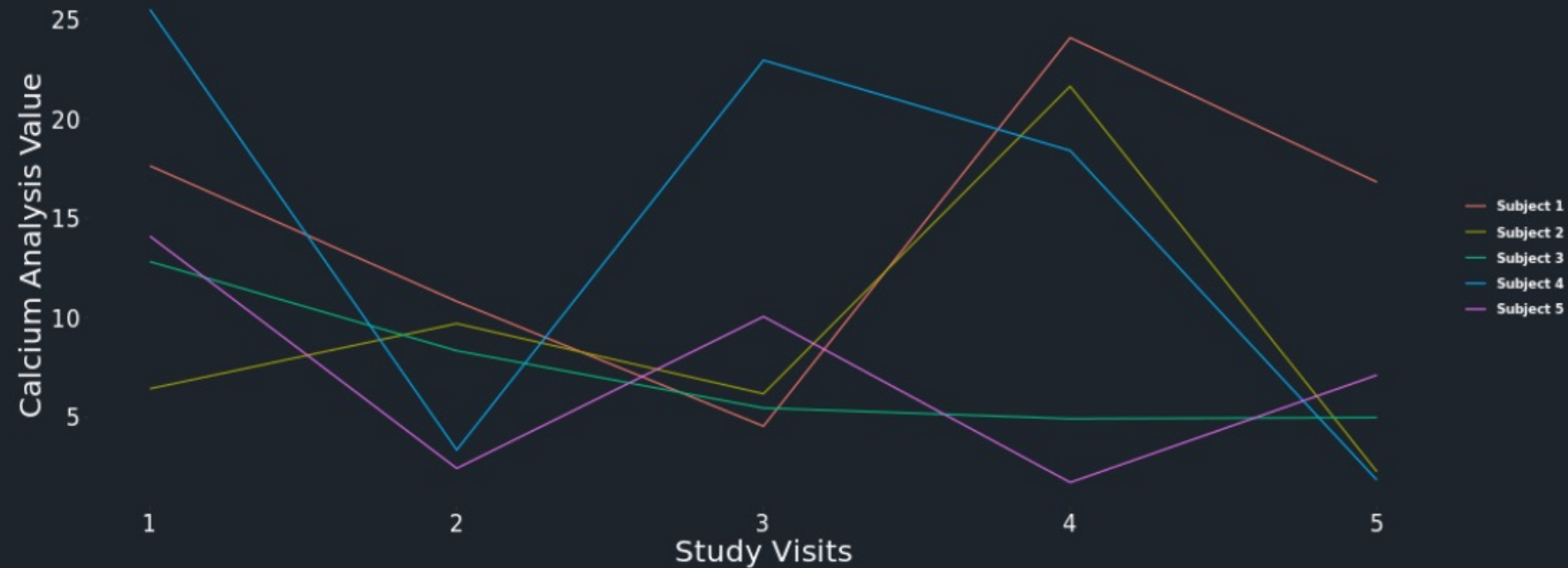
Choose a Parameter Code: dropdown



A Shiny Application

Choose a Parameter Code:

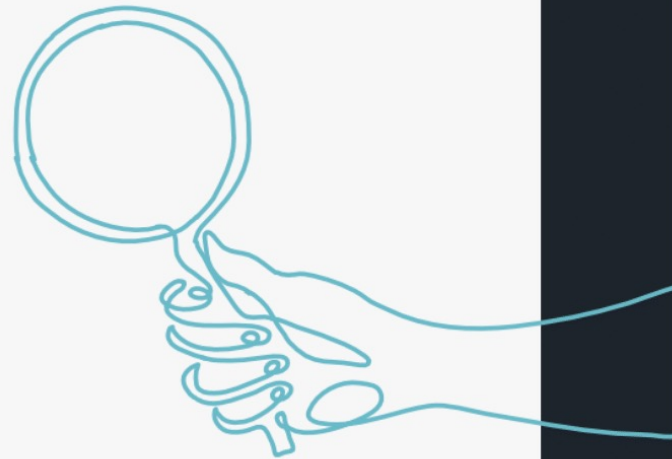
Easily switch
values



Shiny Code

Let's just focus on the select input

```
ui ← fluidPage(  
  selectInput("measurement", NULL, choices = c("Sodium", "Calcium")),  
  plotOutput("plot")  
)  
  
server ← function(input, output, session) {  
  data ← reactive({  
    mydata %>%  
      filter(measurement == input$measurement)  
  })  
  
  output$plot ← renderPlot({  
    ggplot(data(),  
      aes(x = xAxis, y = yAxis,  
          group = IDs, color = IDs)  
    )  
  })  
}  
  
shinyApp(ui, server)
```



Shiny: Magic Under the Hood

HTML

CSS

JavaScript



Shiny: Magic Under the Hood

**Hypertext
Markup
Language**

CSS

JavaScript



Shiny: Magic Under the Hood

HTML

**Cascading
Style Sheets**

JavaScript



Shiny: Magic Under the Hood

HTML

CSS

**The
Language
of the
Internet!**



Shiny: HTML

```
<div class="form-group shiny-input-container">
  <label class="control-label" id="measurement-label" for="measurement">
Y-Axis
  </label>

  <div>
    <select id="measurement">
      <option value="Sodium" selected>Sodium</option>
      <option value="Calcium">Calcium</option>
    </select>
    <script type="application/json" data-for="measurement" data-nonempty="">
      {"plugins":["selectize-plugin-a11y"]}
    </script>
  </div>
</div>
```

One line of code gives us a bunch of HTML markup!

Shiny: CSS

```
.selectize-input {  
  border: 1px solid #cccccc;  
  padding: 6px 12px;  
  display: inline-block;  
  width: 100%;  
  overflow: hidden;  
  position: relative;  
  z-index: 1;  
  -webkit-box-sizing: border-box;  
  -moz-box-sizing: border-box;  
  box-sizing: border-box;  
  -webkit-box-shadow: none;  
  box-shadow: none;  
  -webkit-border-radius: 4px;  
  -moz-border-radius: 4px;  
  border-radius: 4px;  
}
```



We also get styling for free!

Shiny: JavaScript

```
...  
    success: function(res) {  
      $.each(res, function(index, elem) {  
        let optgroupId = elem[settings.optgroupField || "optgroup"];  
        let optgroup = {};  
        optgroup[settings.optgroupLabelField || "label"] = optgroupId;  
        optgroup[settings.optgroupValueField || "value"] = optgroupId;  
        selectize.addOptionGroup(optgroupId, optgroup);  
      });  
      callback(res);  
      if (!loaded) {  
        if (data.hasOwnProperty('value')) {  
          selectize.setValue(data.value);  
        } else if (settings.maxItems === 1) {  
          selectize.setValue(res[0].value);  
        }  
      }  
      loaded = true;  
    }  
  }  
  ...
```

And don't have to worry
about the dropdown's
functionality!

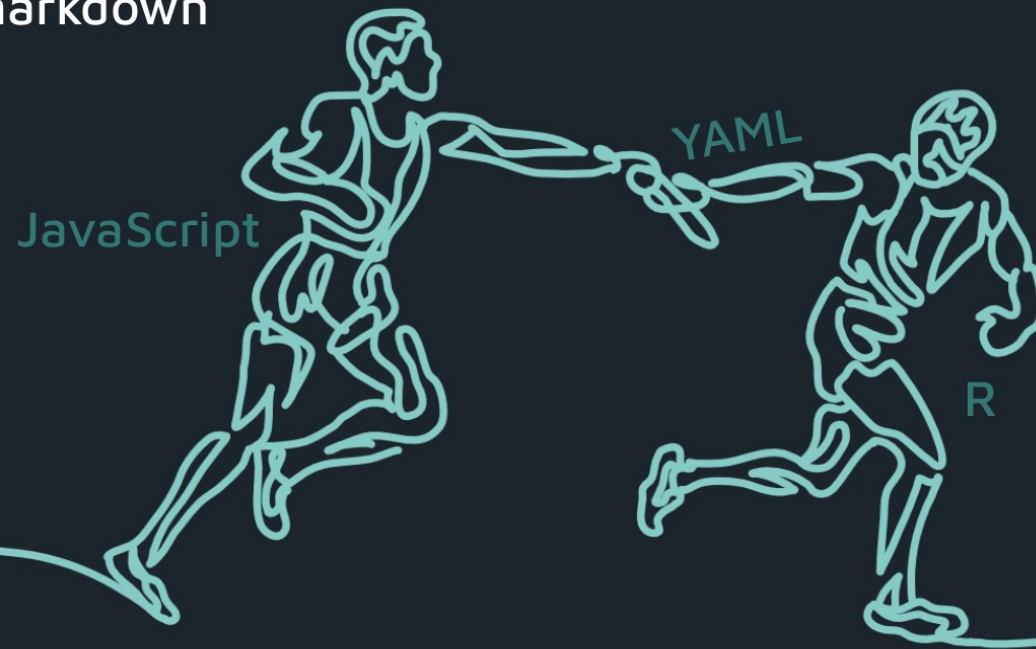
What's an HTMLWidget

Overview

JavaScript runs in the browser

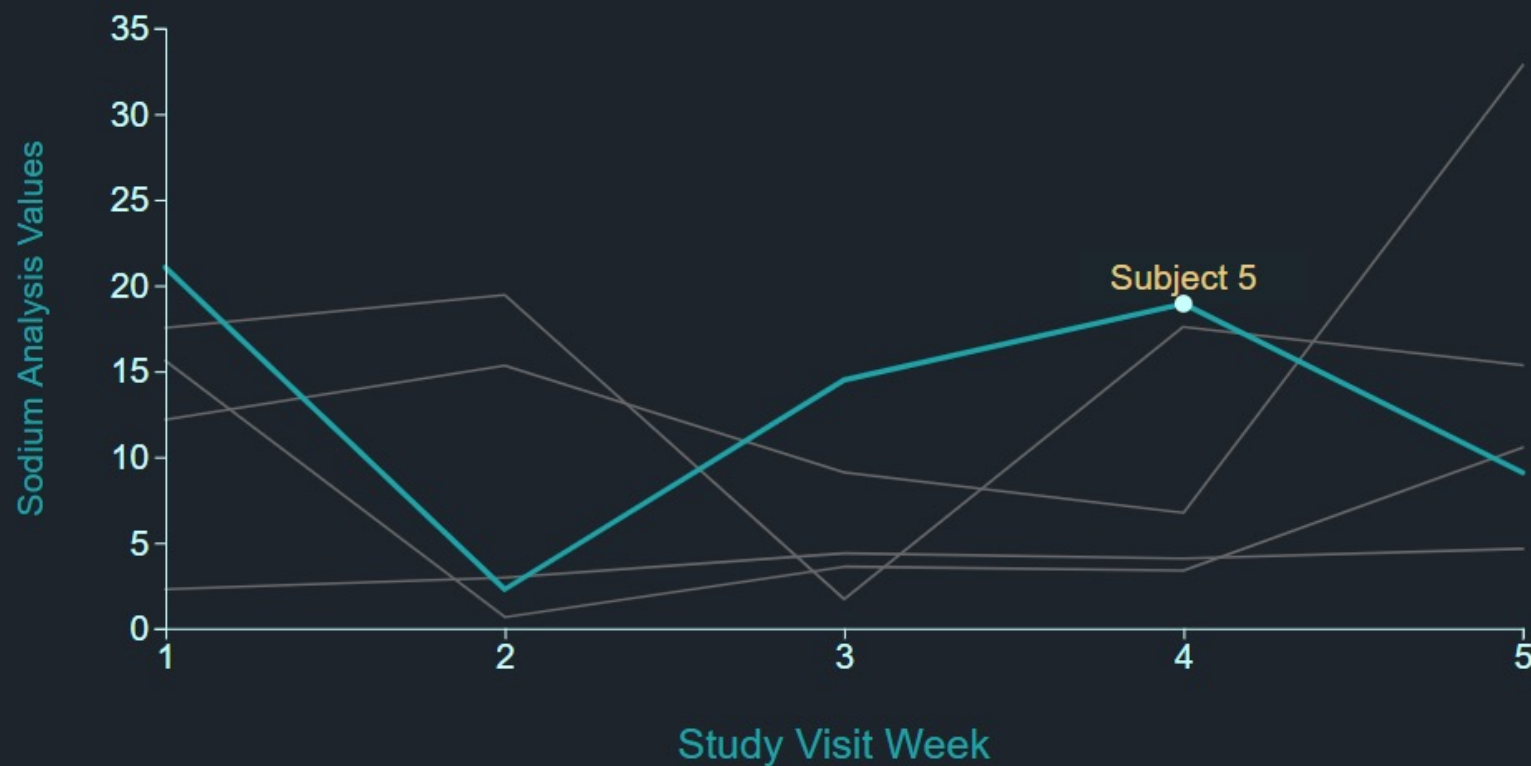
Why reinvent the wheel?

Enhance Shiny and Rmarkdown

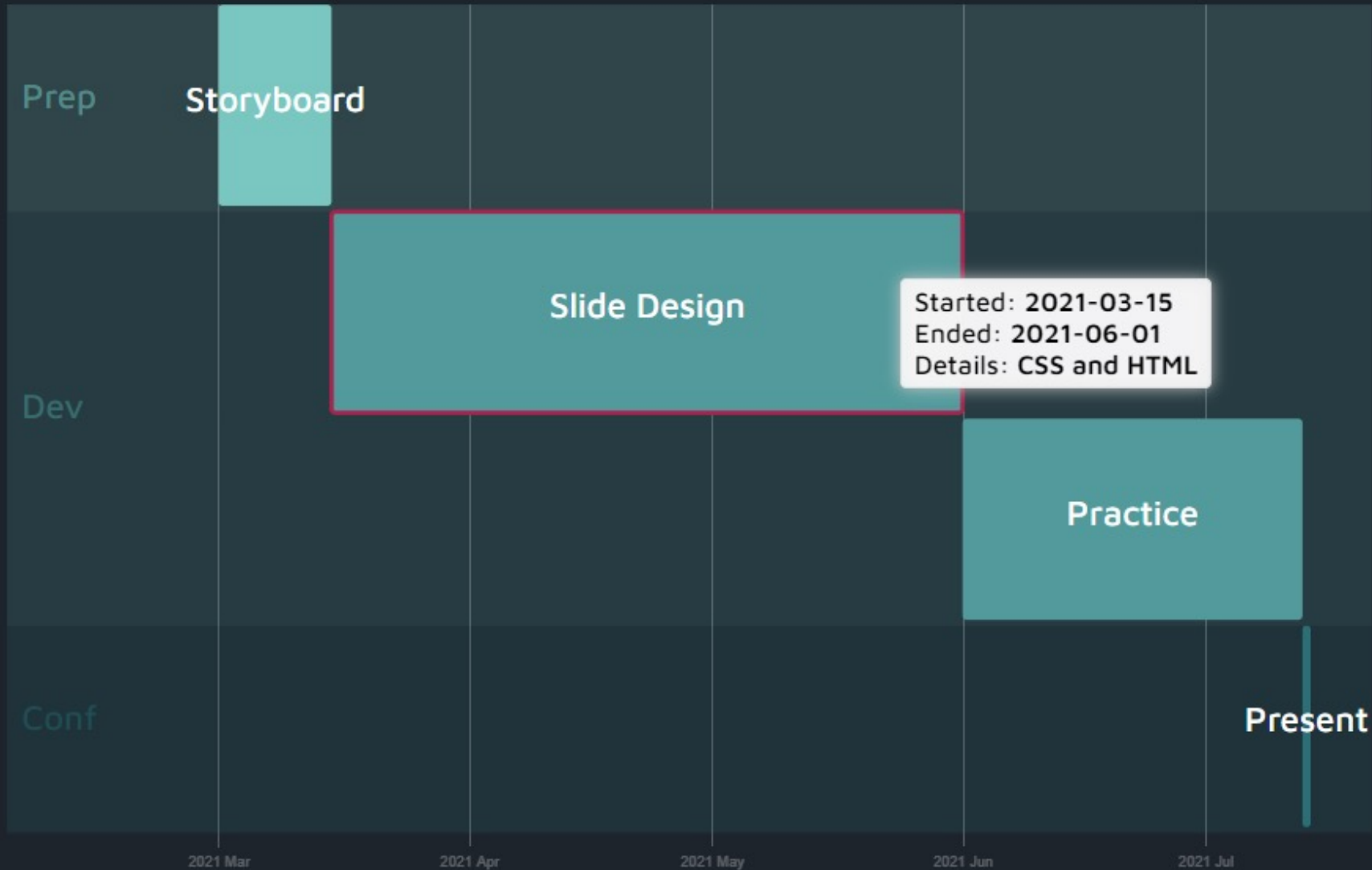


Why should I care?

Example: Hover Line Plot



Further Motivation: Gantt Chart



How do I make one?

Example

```
usethis::create_package("atoruswidgets")
```

```
htmlwidgets::scaffoldWidget("line_select")
```

line_select.R

line_select.yaml

line_select.js

line_select.R

```
#' <Add Title>
#' <Add Description>
#' @import htmlwidgets
#' @export
line_select ← function(message, width = NULL, height = NULL, elementId = NULL) {

  # forward options using x
  x = list(
    message = message
  )

  # create widget
  htmlwidgets::createWidget(
    name = 'line_select',
    x,
    width = width,
    height = height,
    package = 'atoruswidgets',
    elementId = elementId
  )
}
```

→ User supplied string

Wrap into a list

→ Put string in inside HTML element

line_select.yaml

```
# (uncomment to add a dependency)
# dependencies:
#   - name:
#     version:
#     src:
#     script:
#     stylesheet:
```

Custom configuration file

line_select.js

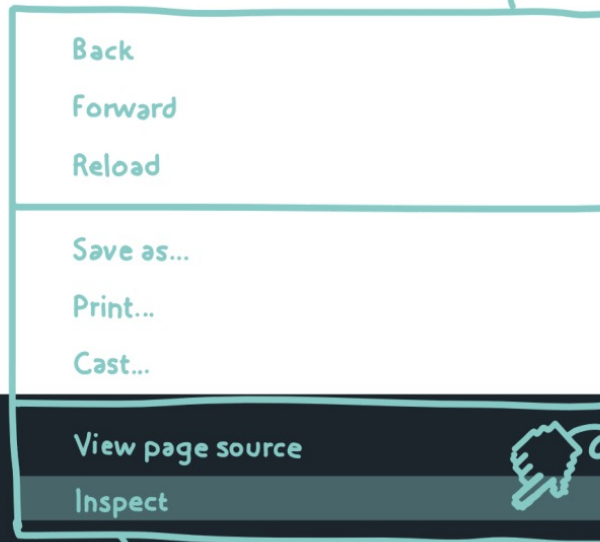
```
HTMLWidgets.widget({  
  name: 'line_select',  
  type: 'output',  
  factory: function(el, width, height) {  
    // TODO: define shared variables for this instance  
    return {  
      renderValue: function(x) {  
        // TODO: code to render the widget, e.g.  
        el.innerHTML = x.message;  
      },  
      resize: function(width, height) {  
        // TODO: code to re-render the widget with a new size  
      }  
    };  
  }  
});
```

kind of like
x\$message

Putting it all together

```
line_plot("This is a line plot.")
```

This is a line plot.



We have a widget!

Under the hood

```
<div id="htmlwidget_container">
  <div
    id="htmlwidget-c21cca0e76e520b46fc7"
    style="width:960px;height:500px;"
    class="line_select html-widget">
    This is a line plot.
  </div>
</div>
<script
  type="application/json"
  data-for="htmlwidget-c21cca0e76e520b46fc7">
  {"x":{"message":"This is a line plot."},"evals":[],"jsHooks":[]}
</script>
```

HTML, CSS, and JavaScript
all created in R

Let Each Language Shine

R is a language and environment for statistical computing and graphics.



D3.js is a JavaScript library for manipulating documents based on data.

Thank you!

Questions?

