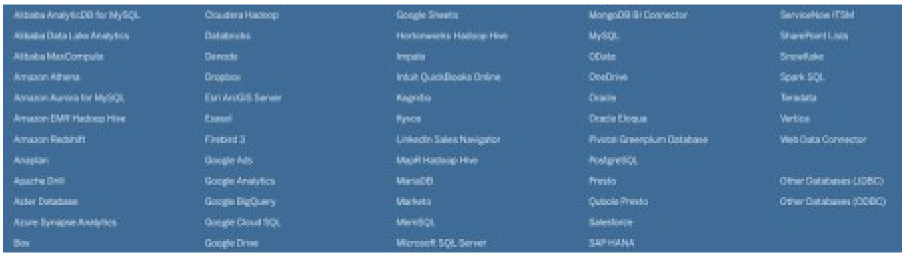
Connectivity

Tableau comes with two options with regards to connecting to data. First, you can connect to a local file –
remote server. The options there are far too many to list, so here's a screenshot of the current offerings (l



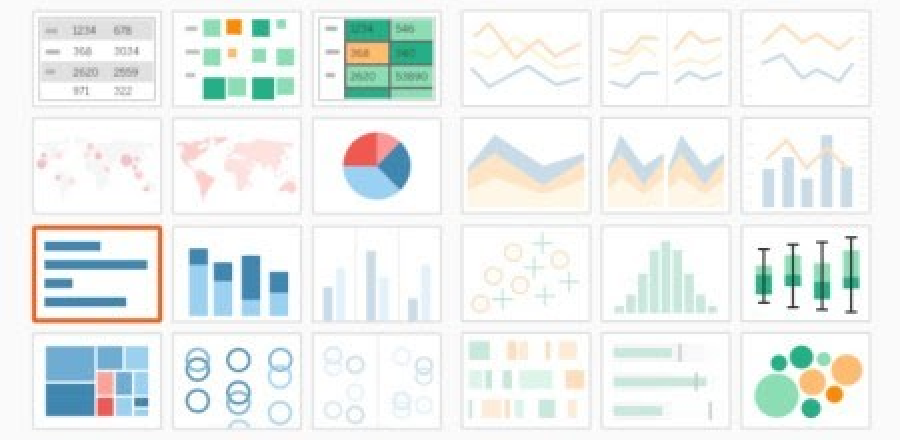
Alibaba AnalyticDB for MySQL
Alibaba Data Lake Analytics
Alibaba MaxCompute
Amazon Athena
Amazon Aurora for MySQL
Amazon EMR Hadoop Hive
Amazon Redshift
Anaplan
Apache Drill
Aster Database
Azure Synapse Analytics
Box
Cloudera Hadoop
Databricks
Denodo
Dropbox
Esri ArcGIS Server
Exasol
Firebird 3
Google Ads
Google Analytics
Google BigQuery
Google Cloud SQL
Google Drive
Google Sheets
Hortonworks Hadoop Hive
Impala
Intuit QuickBooks Online
Kognitio
Kyvos
LinkedIn Sales Navigator
MapR Hadoop Hive
MariaDB
Marketo
MemSQL
Microsoft SQL Server
MongoDB BI Connector
MySQL
OData
OneDrive
Oracle
Oracle Eloqua
Pivotal Greenplum Database
PostgreSQL
Presto
Qubole Presto
Salesforce
SAP HANA
ServiceNow ITSM
SharePoint Lists
Snowflake
Spark SQL
Teradata
Vertica
Web Data Connector
Other Databases (JDBC)
Other Databases (ODBC)

So yeah, there are a lot of options with Tableau. On the other side, R Shiny uses R as the programming l
competitor, the options here are endless. A simple Google search will yield either a premade library or an
can sometimes have an edge. For instance, with R Shiny you can load and analyze gene expression data

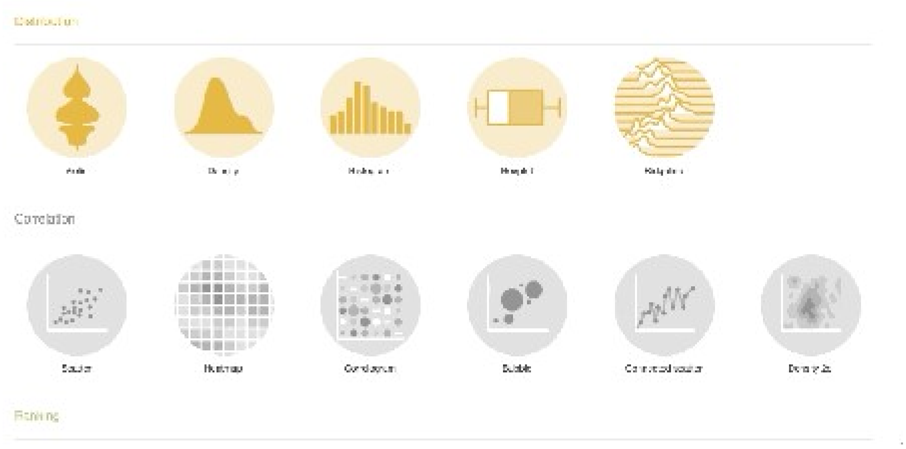
Winner (Connectivity): Tie, with an edge for R Shi

Chart Types

Once again, Tableau doesn't come up short here, with a plethora of visualization options such as bar cha
geographical plots. Refer to the image below for the full list:

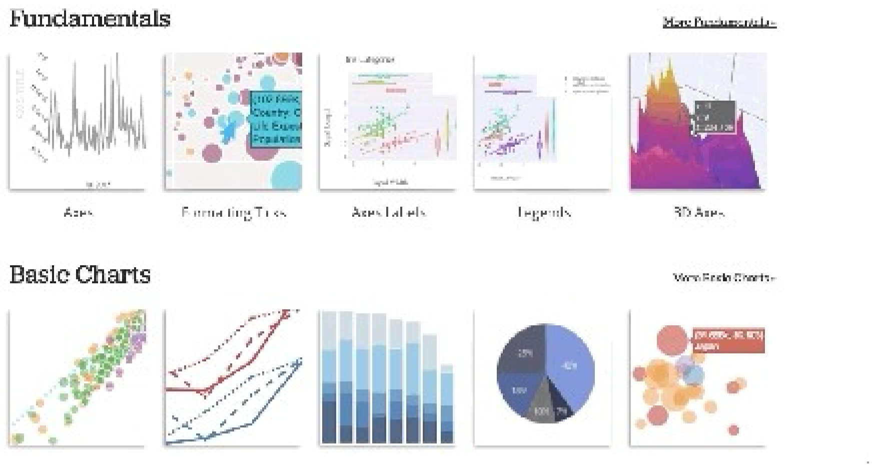


Still, R Shiny demolishes the competition in terms of optionality for chart types. The most widely used visu
visualizations you can make with `ggplot2`:



Source: https://www.r-graph-gallery.com

And here are some cool things that you can do with `Plotly`:



Source: https://plotly.com/r/

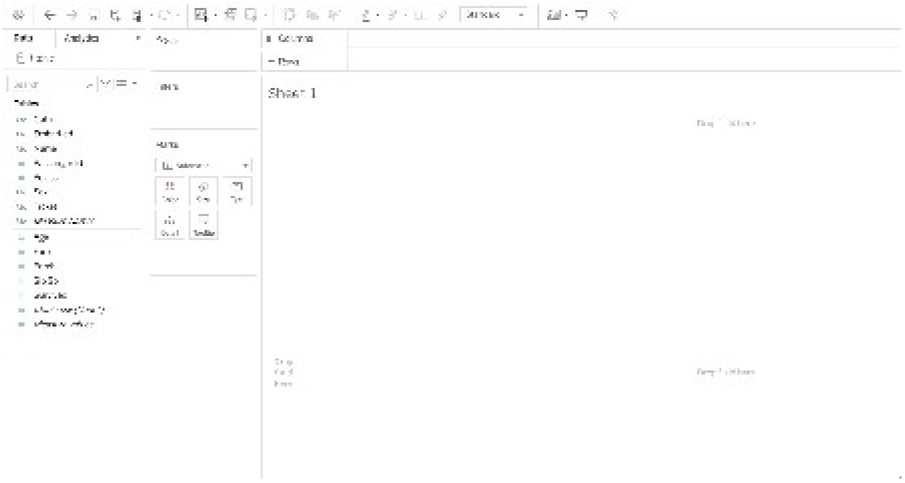
It’s a no brainer that *R Shiny* wins this battle in terms of chart options overall.

Winner (Chart Types): R

Ease of Use: Simple Charts

We’ll now see the extent of the trouble you’ll have to go through to create a simple bar chart, both in Tabl
download the *CSV* file linked here if you’re following along. To be more precise, we’ll create a bar chart c

Let’s start with Tableau. After the data imports, it’s relatively easy to create a visualization. Just follow the



That was very easy, as Tableau was designed to be intuitive and simple to use for non-technical people.

Creating a simple chart is quite a different story with R Shiny. We need to write actual code to get a simpl
data in a concise, visualizable format. Next, we need to introduce some amount of boilerplate to manage
projects, but a real obstacle for someone non-technical to create a simple chart.

Refer to the code below:

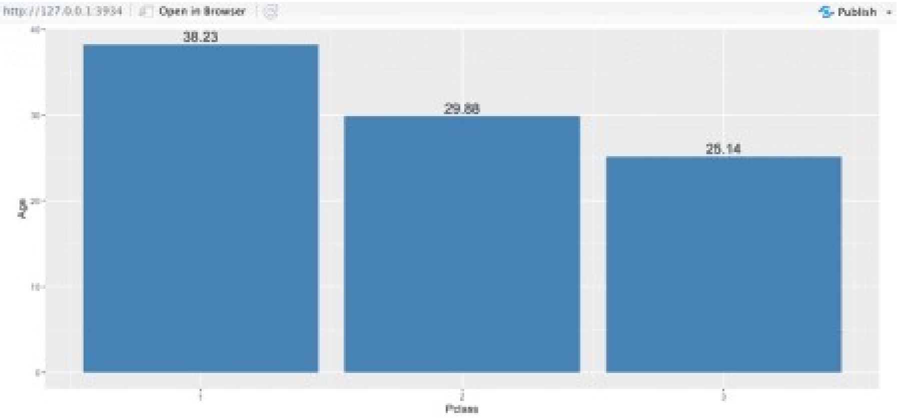
```
library(shiny)
library(ggplot2)

ui <- fluidPage(
    plotOutput('bar')
)

server <- function(input, output) {
    output$bar <- renderPlot({
        pclass_age_data <- titanic %>%
            select(Pclass, Age) %>%
            group_by(Pclass) %>%
            summarise(Age = round(mean(Age, na.rm=TRUE), 2))
        ggplot(data=pclass_age_data, aes(x=Pclass, y=Age)) +
            geom_bar(stat='identity', fill='steelblue') +
            geom_text(aes(label=Age), vjust=-0.3, size=5)
    })
}

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

And here are the results:



As you can see, there’s no point in using *R Shiny* if all you need are simple and straightforward data visu margin. If you still want to use R for charts and you’re willing to put the effort in to have something fully cu simple charts with R can be an onerous task for non-technical people. We recommend Tableau for these

Winner (Simple Charts): T

Ease of Use: Simple Dashboards

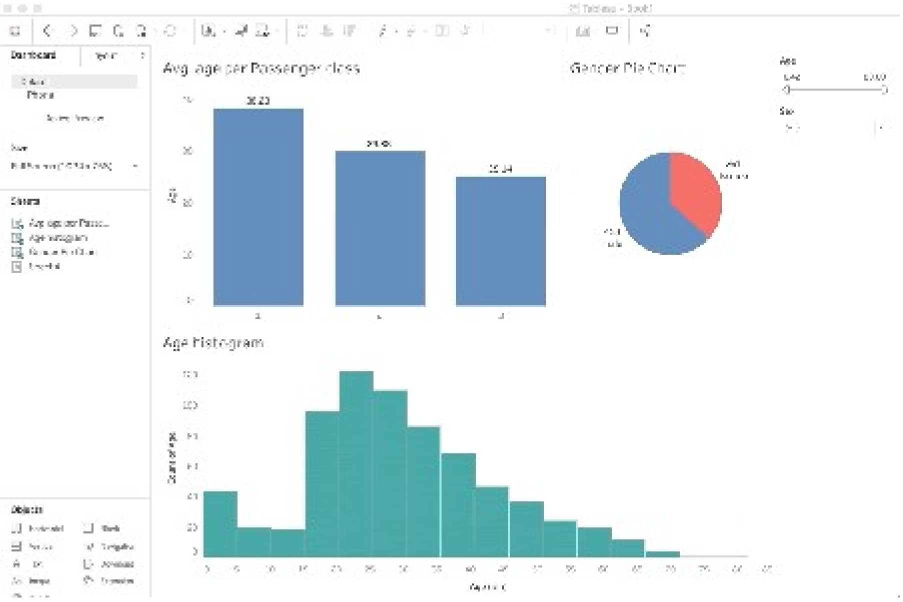
Alright, let’s see who will win in *The Battle of the Dashboards*. We won’t go too complex here, as dashbo charts we’ll put into our dashboard:

- Bar chart – average age per passenger class
- Pie chart – genders
- Histogram – ages

Also, all three charts should update when the user changes one of two filters:

- Age slider – range slider from minimum to the maximum age of the passengers
- Gender picker – option to exclude males or females from the visualizations

Here’s what the final dashboard looks like in *Tableau*:



This entire dashboard required 0 lines of code. We just needed a couple of minutes of clicking in the righ actual R code there. However, it’s not too bad of a deal, since R is relatively easy to learn, and the custo

The following code recreates the dashboard we had in Tableau:

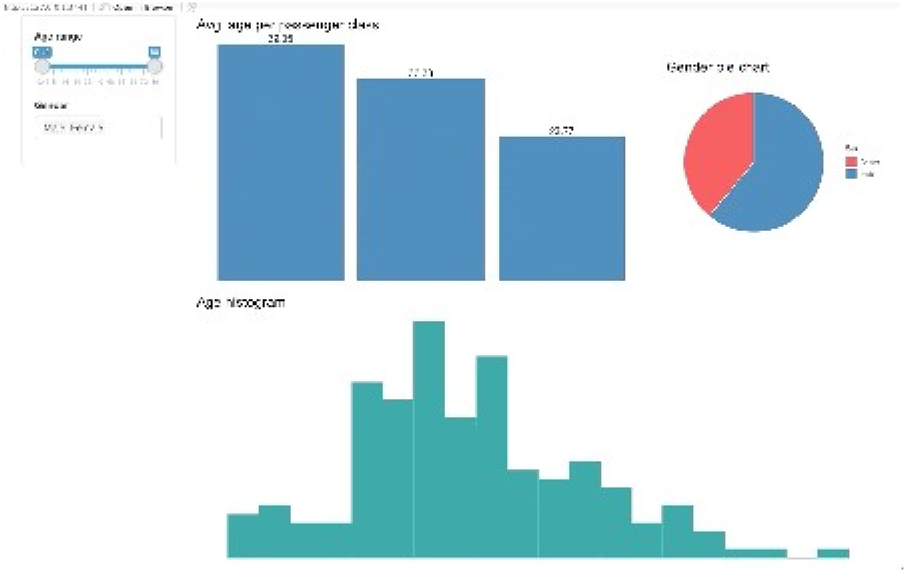
```
library(shiny)
library(dplyr)
library(ggplot2)
library(titanic)
titanic <- titanic::titanic_train

ui <- fluidPage(
    sidebarPanel(
        width = 2,
        sliderInput(
            inputId='ageslider',
            label='Age range',
            min=min(titanic$Age, na.rm=T),
            max=max(titanic$Age, na.rm=T),
            value=c(min(titanic$Age, na.rm=T), max(titanic$Age, na.rm=T))
        ),
        selectInput(
            inputId='genderselect',
            label='Gender',
            choices=c('Male', 'Female'),
            selected=c('Male', 'Female'),
            multiple = TRUE
        )
    ),
    mainPanel(
        fluidRow(
            column(8, plotOutput('bar')),
            column(4, plotOutput('pie'))
        ),
        fluidRow(
            column(12, plotOutput('hist'))
        )
    )
)

server <- function(input, output) {
    output$bar <- renderPlot({
        pclass_age_data <- titanic %>%
            filter(Sex %in% tolower(input$genderselect)) %>%
            select(Pclass, Age) %>%
            filter(Age >= input$ageslider) %>%
            filter(Age <= input$ageslider) %>%
            group_by(Pclass) %>%
            summarise(Age = round(mean(Age, na.rm=TRUE), 2))
        ggplot(data=pclass_age_data, aes(x=Pclass, y=Age)) +
            geom_bar(stat='identity', fill='#4179ab') +
            geom_text(aes(label=Age), vjust=-0.3, size=5) +
            ggtitle('Avg. age per passenger class') +
            theme_void() +
            theme(plot.title=element_text(size=20), panel.background=element_
    })
    output$pie <- renderPlot({
        pie_data <- titanic %>%
            filter(Sex %in% tolower(input$genderselect)) %>%
            filter(Age >= input$ageslider) %>%
            filter(Age <= input$ageslider) %>%
            select(Sex) %>%
            group_by(Sex) %>%
            count() %>%
            mutate(Midpoint = cumsum(n) - n / 2)
        ggplot(pie_data, aes(x='', y=n, fill=Sex)) +
            geom_bar(width=1, stat='identity', color='white') +
            coord_polar('y', start=0) +
            scale_fill_manual(values=c('#f34a53', '#4179ab')) +
            ggtitle('Gender pie chart') +
            theme_void() +
            theme(plot.title=element_text(size=20))
    })
    output$hist <- renderPlot({
        hist_data <- titanic %>%
            filter(Age >= input$ageslider) %>%
            filter(Age <= input$ageslider) %>%
            filter(Sex %in% tolower(input$genderselect)) %>%
            select(Age)
            ggplot(hist_data, aes(x=Age)) +
            geom_histogram(bins=20, color='#62aca8', fill='#1e9a95') +
            ggtitle('Age histogram') +
            theme_void() +
            theme(plot.title=element_text(size=20))
    })
}

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

And here’s what the final product looks like:



As you can see, this dashboard is quite unfinished. That’s what we’ll discuss in the last section of the arti
solid to work in. Tableau’s dashboard is way too sensitive to accidental clicks, and the dashboards look p
much faster with Tableau than with R Shiny. For this reason, we’ll declare Tableau the winner for simple c

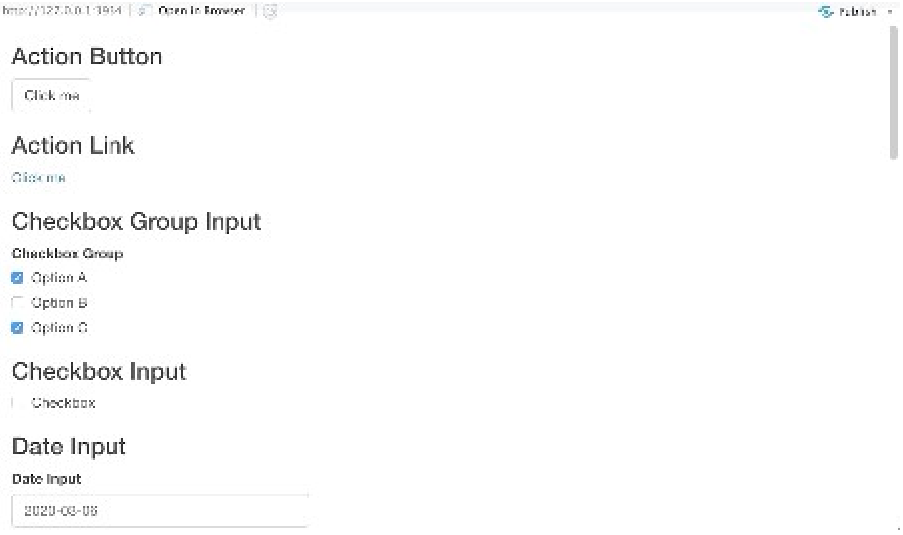
you intend to ratchet up the complexity and visual style of your dashboards.

Winner (Simple Dashboards): Tab

User Input and Interactivity

Here's where things get especially interesting in our Tableau vs. R Shiny comparison. There's simply not for such things. However, inputs are essential for a well-built and interactive dashboard as they provide a Tableau.

R Shiny provides a wide array of input options. Instead of listing them, it's better to show some of them:



Action Button

Click me

Action Link

Click me

Checkbox Group Input

Checkbox Group

☑ Option A

□ Option B

☑ Option C

Checkbox Input

□ Checkbox

Date Input

Date Input

2020-03-08

Yeah, that's a lot, and you can certainly do any task with them. One question that might immediately pop

Well, interactivity makes any dataset much easier to analyze. For example, the *file input* component can further filtered by date (*date range input*) and column selectors (*var select*). This is contrasted with read-c

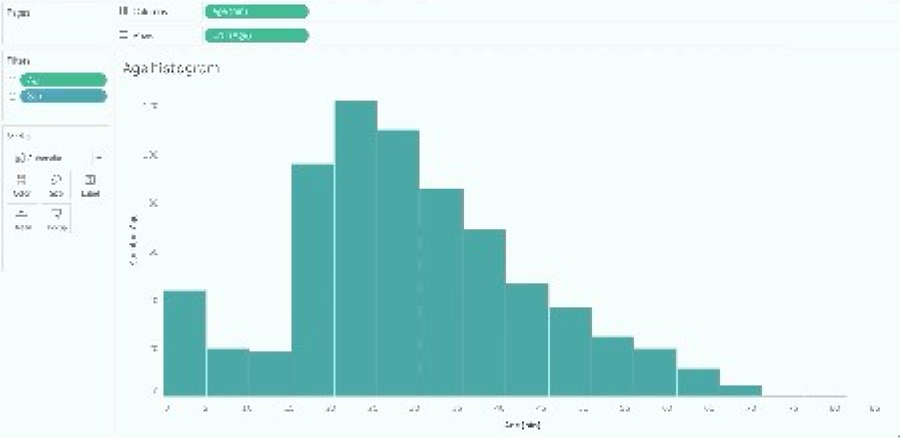
With R Shiny dashboards, you can also build complete interactive forms with password and text field inpu Interactivity is a large part of what separates simple dashboards from powerful, enterprise-level business

Winner (User Input): R S

Visual Styling

Unfortunately, there isn't much you can do to tweak the visuals in Tableau. It's designed to look good out- things. The amount of tweakable aspects depends on the type of plot you're tweaking, but in general, Tab

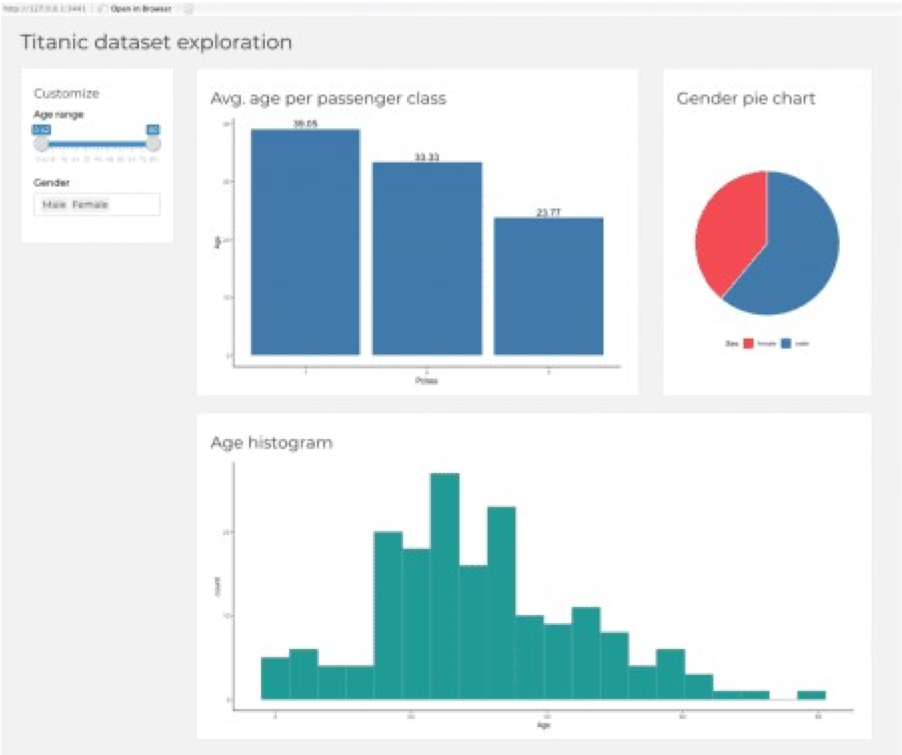
Here's an example of how to change the colors of a histogram:



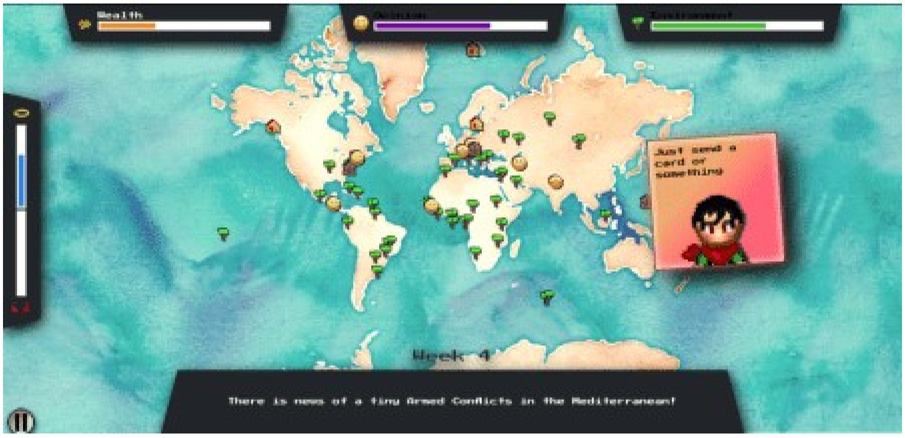
R Shiny is a whole other animal. You can customize everything. To start, create a folder `www` right where
can use everything *CSS* has to offer when it comes to styling R Shiny dashboards.

Learn More: How to Use CSS to Style Your R Shiny Dashboards

To connect the Shiny app to a CSS file, we can put `theme=main.css` right after the `fluidPage` in the F
minutes with CSS and R Shiny:



Here’s the source code for the final styled Shiny dashboard. Give this CSS optionality to a qualified desig
visuals. One of our engineers even created an entire functional card-swiping video game in R Shiny by ta


Wealth
Opinion
Environment
Just send a card or something
Week 4
There is news of a tiny Armed Conflicts in the Mediterranean!

Shiny Decisions is a video game made in R Shiny b

The winner for the visual styling section is once again pretty obvious. R Shiny blows Tableau out of the w
decent out of the box.

Winner (Visual Styling): F

Looking for some attractive Shiny dashboard examples? Find inspiration here in our demo gallery.

Conclusion

Here are the final results, according to our calculations:

- R Shiny – 3 points
- Tableau – 2 points
- Tie – 1 point

So, is Shiny is a clear winner for all situations? Well, no. It depends on the type of work you need to do. If dashboard every now and then, Shiny may be overkill. Tableau was designed for exactly these sorts of ta the actual code, then Tableau is a great choice for simple charts and dashboards.