Creating interactive web graphics often requires a heterogeneous set of technical skills that can include data skills, analytical skills, web development and design. Data needs to be acquired, cleaned, transformed and then sent to a web app for rendering as a plot. This creates a barrier for entry for an analyst to effectively explore data and create interactive graphics in a single language [consistent??] in a flexible, iterative and reproducible way. This chapter will cover how a new **R** package, **ggvis**, lets researchers and analysts communicate results using interactive graphics [needs more]. **ggvis** brings together three ?? ideas: 1) it is based on the *grammar of graphics*, a way to define the elements of graphic; 2) it provides reactive and interactivity plotting in a web-based graphic and; 3) provides support for creating data pipelines in code making graphic creation more comprehensible.

I'll start the chapter by going over the technology suite and dependencies needed to run **ggvis**. I'll also cover the data used in the chapter and where to obtain it. Once the computing and data is covered, I'll introduce how **ggvis** utilizes the *grammar of graphics* to break graphs into components made up of *data*, a *coordinate system*, a *mark type* and *properties*, such as, fill color. This modularity allows analysts to better understand graphical elements and easily swap out different components to make up **n** of plots. Following [more], I'll start to use **ggvis** to build simple static graphs with built-in datasets in R. While demonstrating how to create simple plots, such as, *histograms*, *bar charts*, and *scatter plots*, I'll introduce how **ggvis** incorporates *data pipelines* to improve code

readability. I will also show how **ggvis** employs features from the data manipulation package, **dplyr**, to filter, summarize, or transform the data for in preparation graphing.

After covering the features in **ggvis**, I'll introduce interactivity by plotting library datasets (gate count data, ejournal usage, catalog data). Starting simply, I'll will add a bin slider to a histogram . I will then introduce how to create interactive scatterplots and line graphs with various interactive inputs (a third variable fill color, etc). Finally, I'll introduce and embed our **ggvis** plots in markdown enabled R (Rmarkdown) to create an interactive report. We will also cover, briefly, how to embed and publish out **ggvis** plots in a shiny app.

Limitations

ggvis is still very young and currently not recommended for production use. However, with that said, **ggvis** is under active development and co-authored by two R programmers with a track record of solid R packages.

Conclusion

*ggvis** opens the door for a librarian with knowledge of R to create interactive graphics without knowing the intricacies of a javascript framework or xx. Furthermore, ggvis adoption of both a grammar of graphics and ability to compose multiple pipelines makes it flexible and powerful tool.